МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования

«ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ
И РАДИОЭЛЕКТРОНИКИ» (ТУСУР)

УТВЕРЖДАЮ
Зав. каф. АОИ, д.т.н., проф.
_____ Ю.П. Ехлаков
"____" _____ 2016 г.

МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ВЫПОЛНЕНИЮ ЛАБОРАТОРНЫХ РАБОТ

по дисциплине

"МОДЕЛИРОВАНИЕ И АНАЛИЗ БИЗНЕС-ПРОЦЕССОВ"

для студентов направления подготовки
«Государственное и муниципальное управление»
(бакалавриат)

Часть II

Разработчик:
профессор каф. АОИ, д.т.н.

_____ М.П. Силич

Томск - 2016

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Данное учебно-методическое пособие предназначено для подготовки и выполнения лабораторных работ, включенных во вторую часть лабораторного практикума по дисциплине «Моделирование и анализ бизнес-процессов» для студентов направления "Государственное и муниципальное управление".

Лабораторные работы по данной части дисциплины имеют **целью**: закрепление теоретического материала, получение навыков самостоятельного имитационного анализа с помощью инструментального средства Arena.

Выполнение лабораторных работ направлено на формирование следующей **компетенции**:

- умение моделировать административные процессы и процедуры в органах государственной власти Российской Федерации, органах государственной власти субъектов Российской Федерации, органах местного самоуправления, адаптировать основные математические модели к конкретным задачам управления (ПК-7).

Лабораторные работы выполняются индивидуально. Лабораторная работа выполняется в соответствии с порядком, описанном в методических указаниях.

Форма контроля выполнения лабораторной работы: демонстрация преподавателю результатов работы, собеседование, ответы на вопросы, выполнение дополнительных заданий.

Лабораторная работа №*1*
«Построение имитационной модели Arena»

Цель работы: Ознакомиться с основными возможностями языка моделирования SIMAN и основами работы с системой имитационного моделирования Arena 13.5. Создать модель бизнес-процесса, заданного в качестве индивидуального задания, с помощью инструментального средства Arena.

Порядок выполнения работы.

1. Выбор задания.

Выберите бизнес-процесс, для которого будете формировать имитационную модель. Вы должны выбрать один из вариантов процессов, описанных в приложении 1. Порядок выполнения работы будет иллюстрироваться на примере моделирования работы салона красоты, Вы же должны выполнять все этапы построения модели для выбранного бизнес-процесса.

2. Знакомство с основами имитационного языка моделирования SIMAN

SIMAN, впервые реализованный в 1982 г., – чрезвычайно гибкий и выразительный язык имитационного моделирования, постоянно совершенствующийся путем добавления в него новых возможностей.

Имитационное моделирование дает возможность изучать объекты, о поведении которых имеется недостаточно информации. SIMAN позволяет анализировать ход выполнения процессов, выявлять "узкие места" в потоках работ. При этом можно рассматривать процессы в различных масштабах времени, отслеживать переменные, наиболее важные для успешного функционирования моделируемой системы, и анализировать имеющиеся связи между различными переменными.

Также имитационное моделирование позволяет проверять гипотезы о причинах возникновения тех или иных наблюдаемых феноменов, помогает получить ответ на вопрос "что, если...". Несомненным преимуществом является возможность опробовать изменения в ходе выполнения бизнес-процессов без дорогостоящих «натурных» экспериментов. Новая политика, управляющие процедуры, правила принятия решений, организационная структура, потоки информации и т.д. могут быть исследованы без вмешательства в работу реальной системы. Новые технические средства, планы размещения, программное обеспечение, транспортные системы и т. п. могут быть опробованы до того, как деньги, время и другие ресурсы будут потрачены на их приобретение и/или создание.

К сожалению, несмотря на неоспоримые достоинства имитационного моделирования, в настоящее время в России этот метод исследования сложных систем используется крайне редко. Это связано с тем, что разработка таких моделей требует больших временных и стоимостных затрат. Но тенденции последнего времени вселяют надежду на то, что ситуация изменится и имитационное моделирование в России будет также широко и активно использоваться, как в США, Канаде и Европе. Именно для того чтобы компенсировать этот пробел российской действительности, средства имитационного моделирования рассматриваются на примере мощного программного пакета Arena 13.5.

3. Знакомство с основными возможностями Arena 13.5

Программное средство Arena, разработанное компанией Systems Modeling Corporation для имитационного моделирования, позволяет создавать динамические компьютерные модели, используя которые можно адекватно представить очень многие реальные системы. Самая первая версия этой системы увидела свет в 1993 г. Arena снабжена удобным объектно-ориентированным интерфейсом и обладает удивительными возможностями по адаптации к всевозможным предметным областям (рис. 1.1).

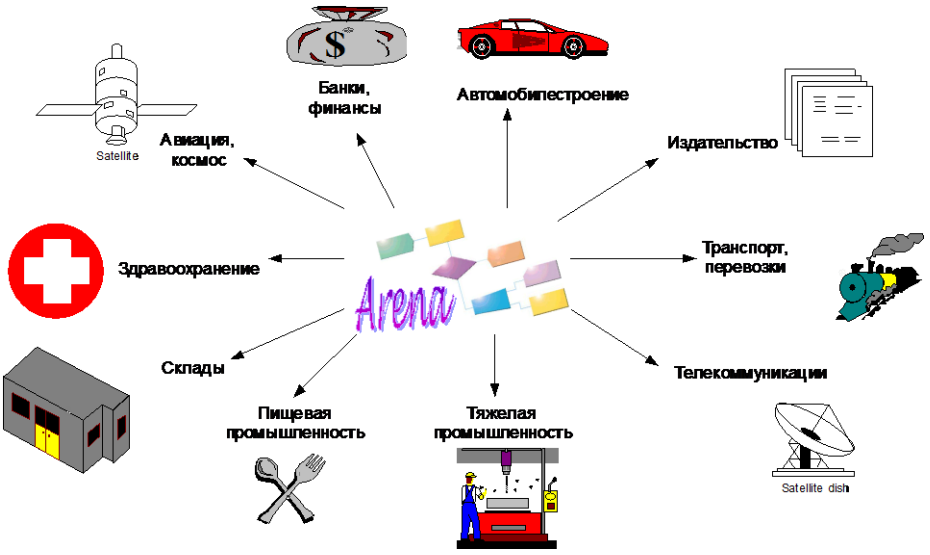


Рис. 1.1. Области применения Arena

Математическим базисом программного пакета Arena является теория систем массового обслуживания. Основа технологий Arena – язык моделирования SIMAN. Для отображения результатов моделирования используется анимационная система Cinema animation, известная на рынке с 1984 г. В целом система Arena исключительно проста в использовании. Процесс моделирования организован следующим образом: сначала пользователь шаг за шагом строит в визуальном редакторе системы Arena модель, затем система генерирует по ней соответствующий код на SIMAN, после чего автоматически запускается Cinema animation.

Интерфейс Arena включает в себя всевозможные средства для работы с данными, в том числе электронные таблицы, базы данных, ODBC, OLE, поддержку формата DXF.

С Arena пользователь сможет:

- моделировать процессы для последующего исследования, документирования, коммуникаций;
- моделировать бизнес «as to be» («как должно быть»), отразить все протекающие процессы, определить возможности усовершенствования;
- визуализировать процессы с помощью динамической графики и мультипликации;
- анализировать эффективность системы «as is» («как есть») и сравнивать большое число альтернатив «as to be» («как должно быть») для выбора наилучшей.

4. Запуск Arena 13.5.

После запуска Arena автоматически открывается новый файл. Чтобы сохранить модель необходимо выбрать File → Save. Выбрав каталог, ввести название модели и нажать кнопку «Сохранить».

Среда моделирования Arena представлена на рис 1.2.

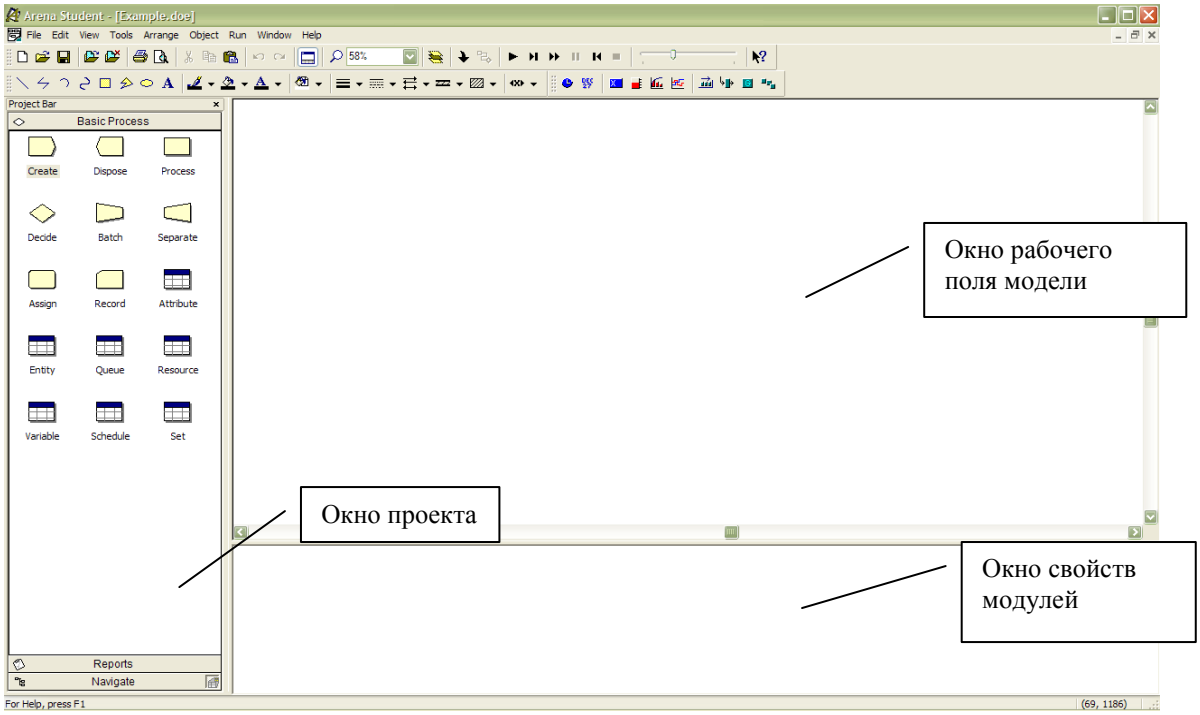


Рис. 1.2. Среда моделирования Arena

Окно приложения разделено на три области:

1. Окно рабочего поля модели, в котором описывается логика модели с использованием схемных (графических) модулей. Окно рабочего поля представляет графику модели, включая блок-схему процесса, анимацию и другие элементы.

2. Окно свойств модулей, в котором отображаются свойства всех модулей (как модулей данных, так и схемных), имеющихся и используемых в модели.

3. Окно проекта – это навигатор системы, в котором отображается рабочая панель со всеми модулями и другие доступные и открытые панели.

Окно проекта включает в себя несколько панелей:

- Basic Process Panel (панель основных процессов) – содержит модули, которые используются для моделирования основной логики системы.

- Advanced Process Panel (панель усовершенствованных процессов) – содержит дополнительные модули для создания моделей со сложной логикой процесса.

- Advanced Transfer Panel (панель перемещения) – содержит специально разработанные блоки для моделирования процесса перемещения объектов с помощью транспортера или конвейера.

- Reports (панель отчетов) – панель сообщений: содержит сообщения, которые отображают результаты имитационного моделирования.

- Navigate (панель навигации) – панель управления позволяет отображать все виды модели, включая управление через иерархические подмодели.

Рассмотрим состав панели основных процессов Basic Process Panel (рис.1.3.). Эта панель состоит из двух типов модулей: схемных модулей и модулей данных. **Схемные модули (Flowchart Modules)** кратко описаны в таблице 1.1.

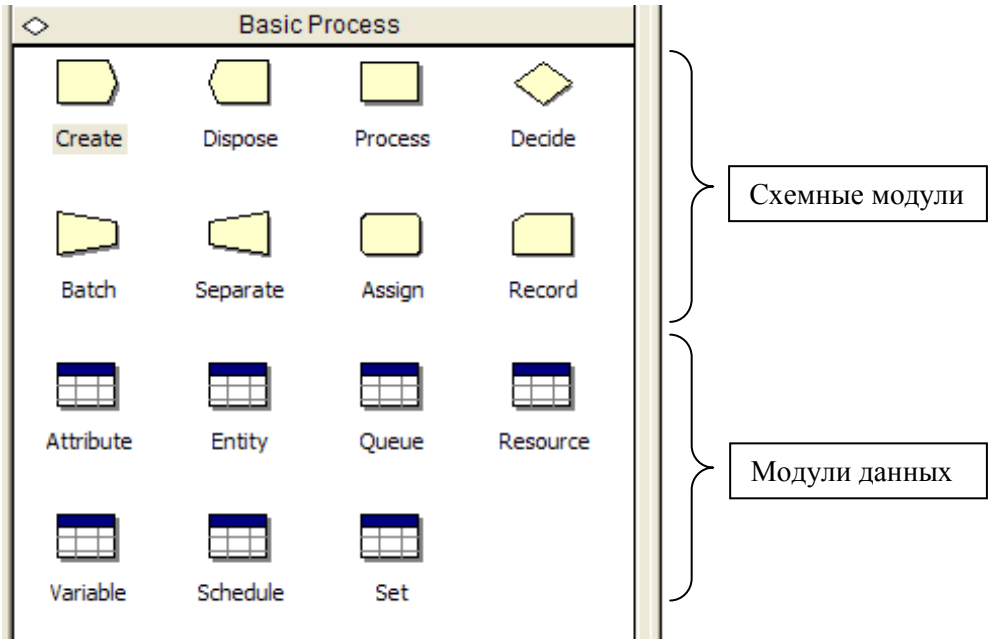
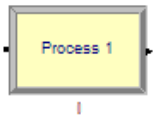
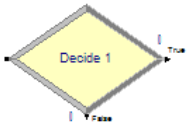
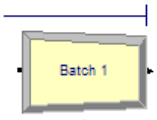
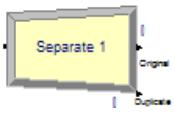
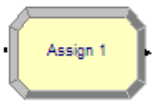
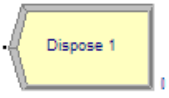


Рис. 1.3. Типы модулей панели Basic Process

Таблица 1.1

	Модуль Create - создает сущности, обрабатываемые в системе.
	Модуль Process - имитирует процесс обработки сущностей.
	Модуль Decide позволяет проверять условия и в зависимости от результата проверки направлять сущности тому или иному процессу.
	Модуль Batch отвечает за механизм группировки сущностей в имитационной модели.
	Модуль Separate может использоваться для создания копий и для разделения ранее сгруппированных сущностей.
	Модуль Assign предназначен для задания значения атрибута сущности.
	Модуль Record предназначен для сбора статистики в имитационной модели.
	Модуль Dispose удаляет сущности из системы.

Подробное описание схемных модулей приводится в приложении 2.

Все **модули данных (Data Modules)** в навигаторе панелей имеют одинаковый вид, т. к. они не отображаются физически в блок-схеме модели, в связи с этим их изображение не приводится.

Модуль Entity определяет тип сущности и ее анимационную картинку в имитационном процессе, также определяет стоимостную информацию.

Модуль Queue предназначен для изменения правила расстановки сущностей в очереди.

Модуль Resource предназначен для определения ресурсов и их свойств в имитационном процессе.

Модуль Schedule может использоваться вместе с модулем Resource для определения вместимости ресурса и с модулем Create – для задания расписания прибытия сущностей.

Модуль Set описывает группу ресурсов, использующихся в модуле Process.

Модули Variable и Attribute определяют значение переменных.

Подробное описание модулей содержится в приложении 2.

5. Использование модуля Create.

Построение модели начинается с модуля Create. Это отправная точка для входящих компонентов модели.

Рассмотрим использование модуля создания сущностей на примере работы салона красоты. Допустим, в салон приходят клиенты трех типов. Клиенты первого типа желают только стричься. Распределение интервалов их прихода 35±10 мин. Клиенты второго типа желают постричься и сделать маникюр. Распределение интервалов их прихода 60±20 мин. Клиенты третьего типа желают только сделать маникюр. Распределение интервалов их прихода 50±10 мин.

Перетащим модуль Create из панели модулей в окно модели три раза, т. к. в салон красоты приходят три типа клиентов (на стрижку, на маникюр и на стрижку и маникюр). Название модулей по умолчанию «Create1», «Create2», «Create3» (рис. 1.4).

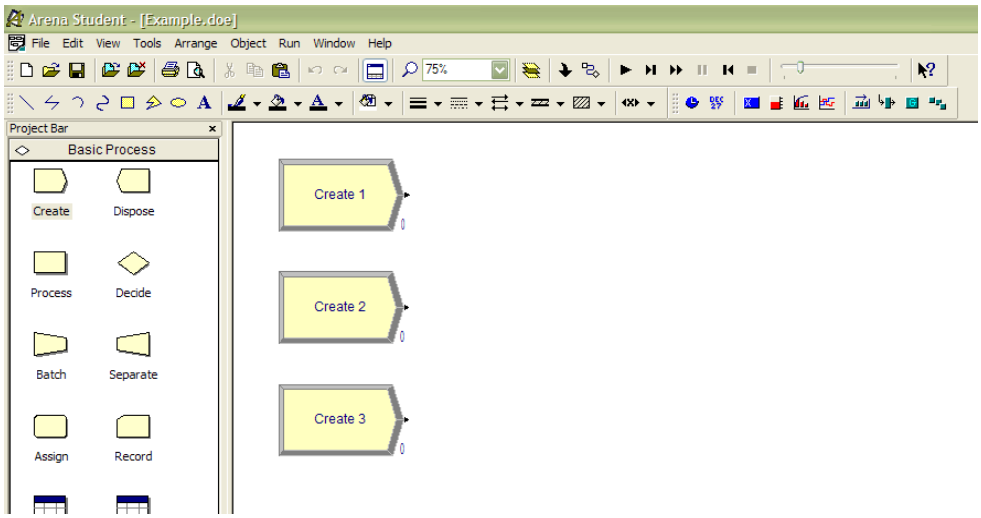


Рис. 1.4. Помещение модулей Create

Модуль «Create1» отвечает за создание клиентов первого типа (желающих только постричься). Чтобы определить свойства этого модуля, нужно щелкнуть по нему левой кнопкой мыши двойным щелчком. Появится диалог (пример показан на рис. 1.5).

В появившемся диалоге переименуем модуль «Create1» в «Na strigku». Параметру Type присвоим значение Expression, т. к. поток прибытия будет формироваться по определенному выражению. Поскольку распределение интервалов прихода клиентов первого типа – 35±10 мин., то параметру Expression присвоим значение UNIF (Min, Max), где min=35-10=25 и max=35+10=45. Значение параметру Units выбираем Minutes, т.к. распределение интервалов прихода клиентов измеряется в минутах.

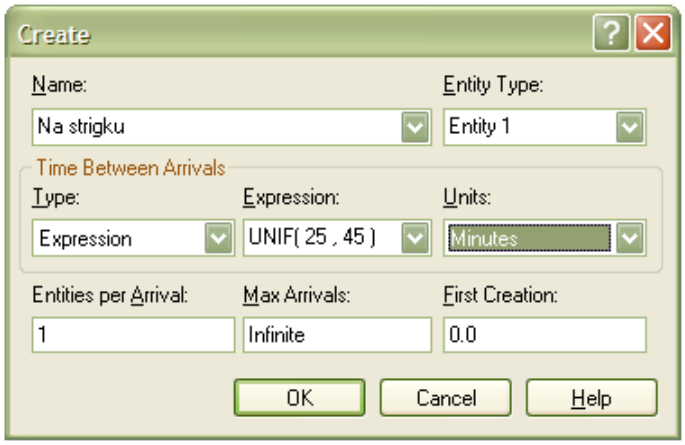


Рис. 1.5. Задание свойств модуля «Create1»

Аналогично опишем модули «Create2» и «Create3». Результаты приведены на рисунках ниже (рис. 1.6 и рис. 1.7).

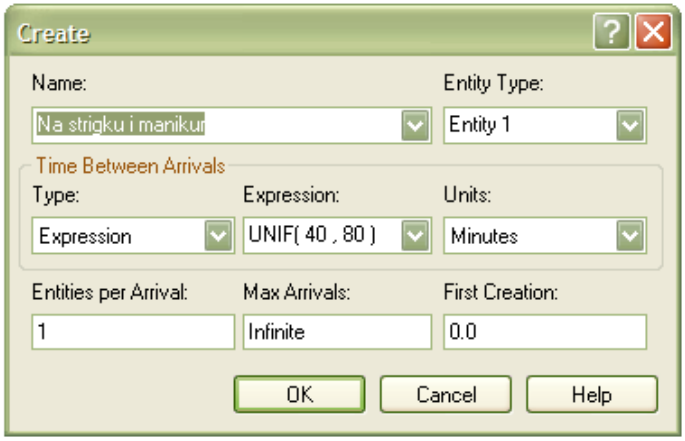


Рис. 1.6. Задание свойств модуля «Create2»

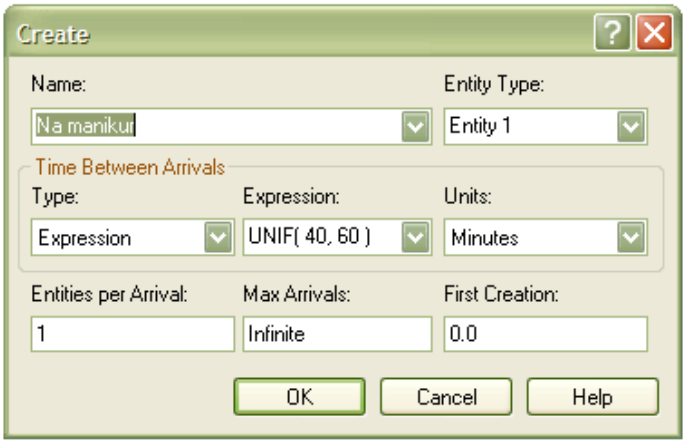


Рис. 1.7. Задание свойств модуля «Create3»

6. Использование модуля Assign.

Для того чтобы управлять сущностями, создаваемыми модулями Create, введем два атрибута с булевыми значениями. Первый атрибут будет отвечать за желание клиента подстричься (т.е. значение атрибута в этом случае равно 1), а второй – за желание клиента сделать маникюр. Для этого воспользуемся модулем Assign, т.к. именно этот модуль предназначен для задания нового значения переменной, атрибуту сущности, типу сущности и т.д.

Перетащим модуль Assign из панели модулей в окно модели три раза, чтобы задать атрибуты для клиентов, порождаемых каждым из трех модулей Create. Название по умолчанию «Assign1», «Assign2», «Assign3» (рис. 1.8).

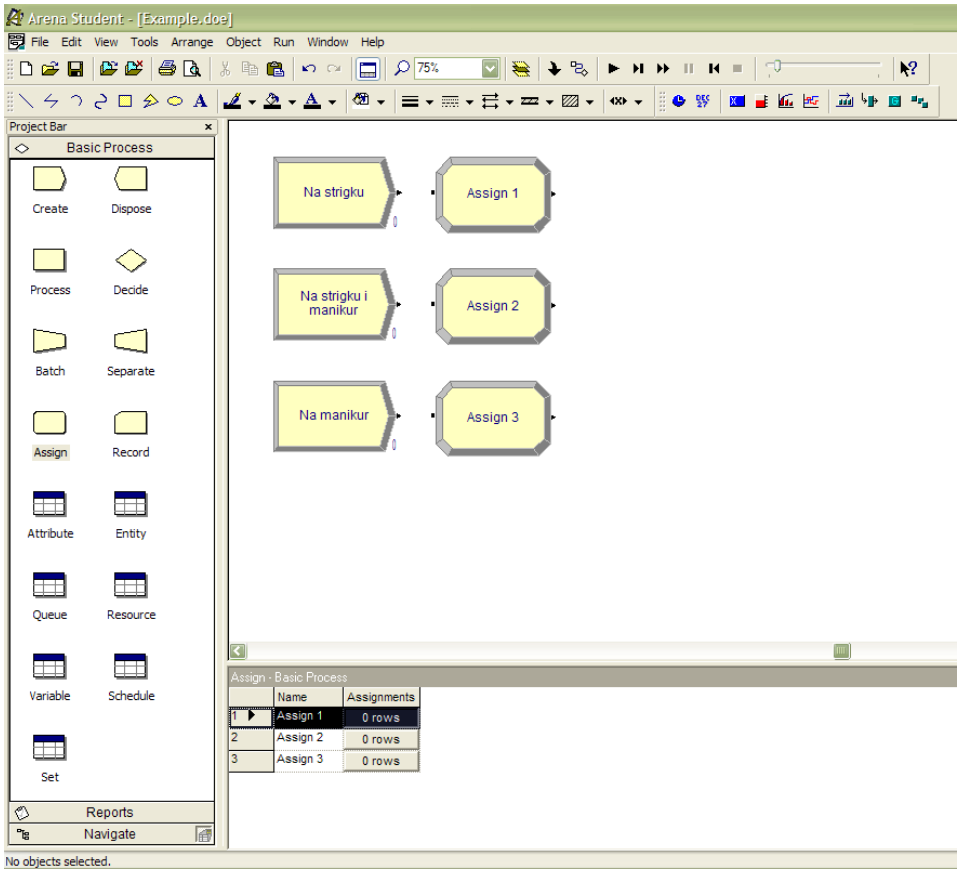


Рис. 1.8. Помещение модулей Assign

Щелкнем два раза по первому модулю «Assign1», появится окно задания свойств модуля (рис. 1.9).

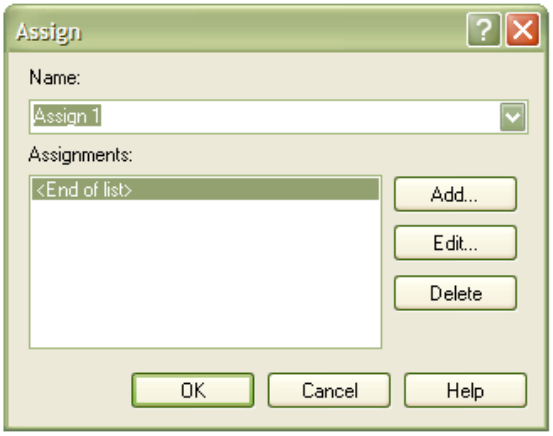


Рис. 1.9. Окно свойств модуля Assign

Теперь добавим первый атрибут, нажав кнопку «Add…». Параметру Type присваиваем значение Attribute, параметру Attribute Name – Attribute 1, а значение атрибута (параметр New Value) – 1, т.к. у клиентов первого типа имеется желание подстричься (рис. 1.10).

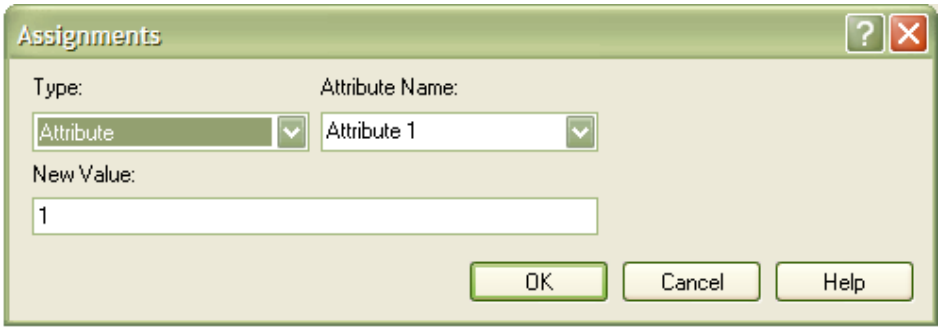


Рис. 1.10. Добавление атрибута

Аналогично добавим второй атрибут. Параметр Attribute Name будет иметь значение Attribute 2, а параметр New Value – 0, т.к. клиенты этого типа пришли только на стрижку и не желают делать маникюр. В результате получим следующее описание атрибутов модуля «Assign1» (рис. 1.11):

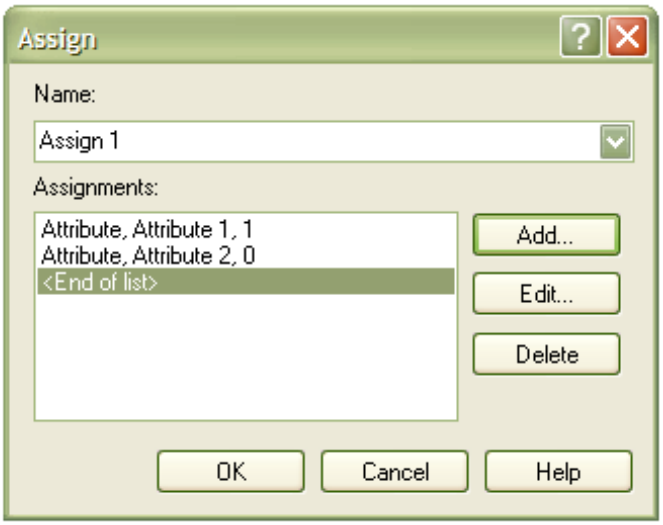


Рис. 1.11. Задание свойств модуля «Assign1»

Таким же образом зададим атрибуты модулей «Assign2» и «Assign3». Модуль «Assign2» будет иметь следующие значения атрибутов: Attribute 1 – 1, Attribute 2 – 1, т.к. клиенты второго типа желают и подстричься, и сделать маникюр. Атрибуты модуля «Assign3» будут следующими: Attribute 1 – 0, Attribute 2 – 1, т.к. клиенты третьего типа желают только сделать маникюр.

Осталось соединить модули: «Assign1» с модулем «Create1», «Assign2» с «Create2», «Assign3» с «Create3». Для этого воспользуемся коннектором  , который находится на стандартной панели управления.

Получившаяся модель приведена на рис. 1.12.

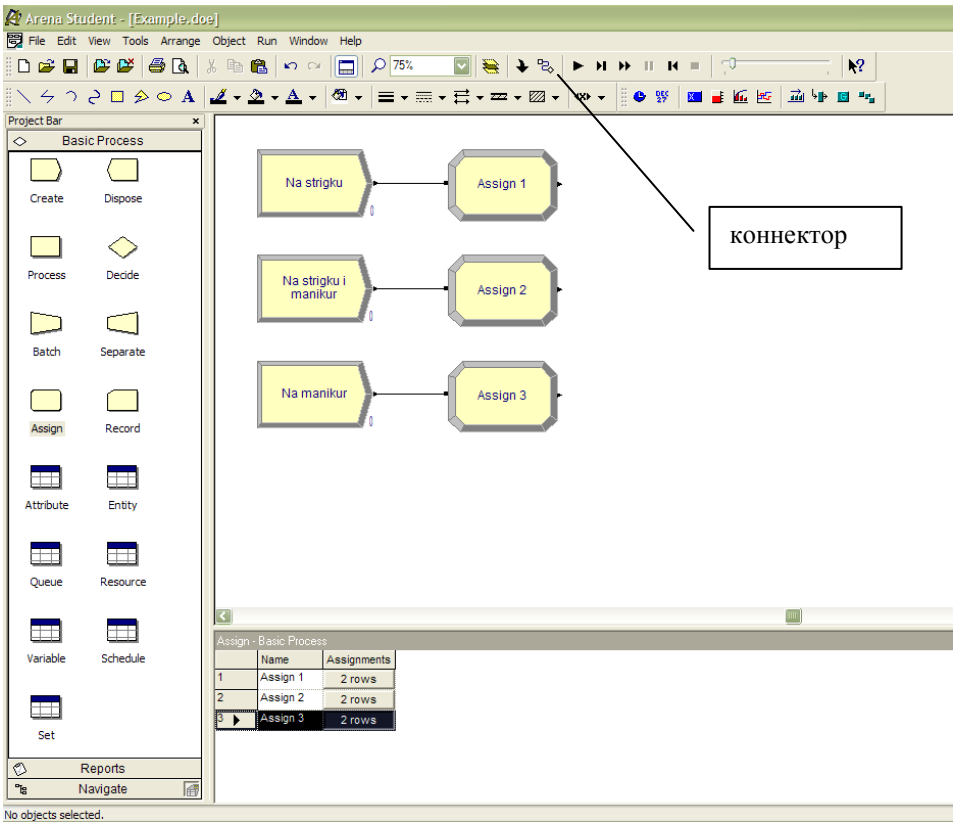


Рис. 1.12. Соединение модулей Assign и Create

7. Использование модуля Process.

Следующим шагом в построении модели будет направление сущностей на обработку.

В нашем примере клиенты первого типа направляются к парикмахеру. Чтобы отобразить процесс стрижки в модели, воспользуемся модулем Process. Именно этот модуль отвечает за процесс обработки сущностей в имитационной модели.

Перетащив модуль Process из панели модулей в окно модели и щелкнув по нему два раза, опишем его по условиям задания (рис. 1.13). В поле Name задается имя модуля, например, «Strigka». В качестве значения параметра Type выбираем Standard, т.к. логическая схема выполнения процесса определяется внутри модуля (через параметр Action). Для параметра Action выбираем значение Seize Delay Release, т.к. при выполнении процесса ресурс (парикмахер) занят определенное время. Значением параметра Priority выбираем Medium (2), т.е. приоритет этого модуля при использовании общего для нескольких процессов ресурса средний. В поле Resources определяем используемые модулем ресурсы. Добавить ресурс можно нажав кнопку «Add…». Задаем имя ресурса, например, Resources1 (ресурсы создаются аналогично атрибутам в модуле Assign).

В поле Delay Type выбираем тип задания времени обработки – Expression, т. к. время обработки задается выражением. Для параметра Units задаем значение Minutes, т. к. процесс измеряется в минутах. В поле Allocation выбираем Non-Value Added, т .к. стоимостные характеристики не будем учитывать. В поле Expression задаем выражение, определяющее значение временной задержки. Допустим, на стрижку уходит 30±15 мин. Тогда вводим выражение: UNIF (15, 45).

Подробное описание параметров модуля Process (и других модулей) приведено в приложении 2.

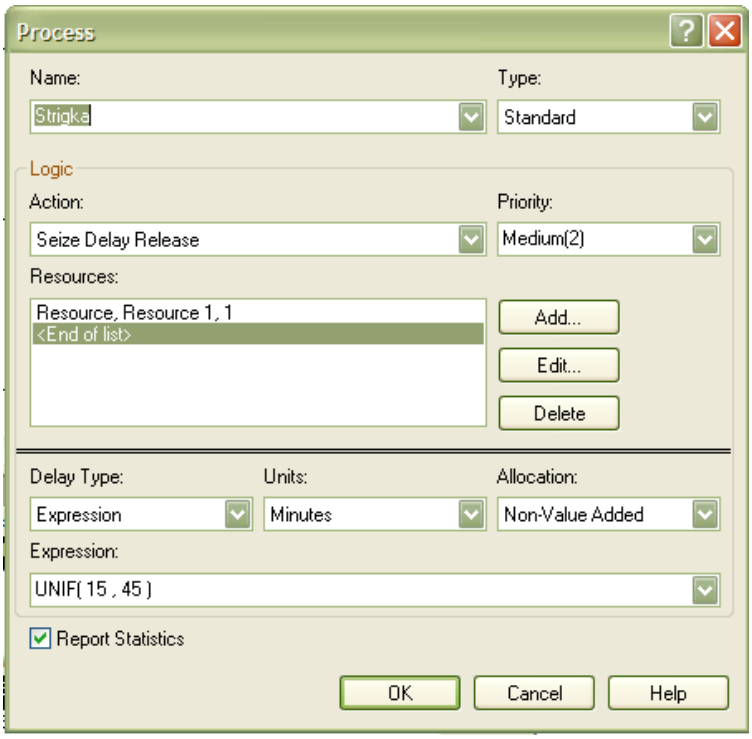


Рис. 1.13. Задание свойств модуля «Strigka»

Нам осталось в окне модели соединить модуль «Assign1» с модулем «Strigka», воспользовавшись коннектором. Результат приведен на рис. 1.14.

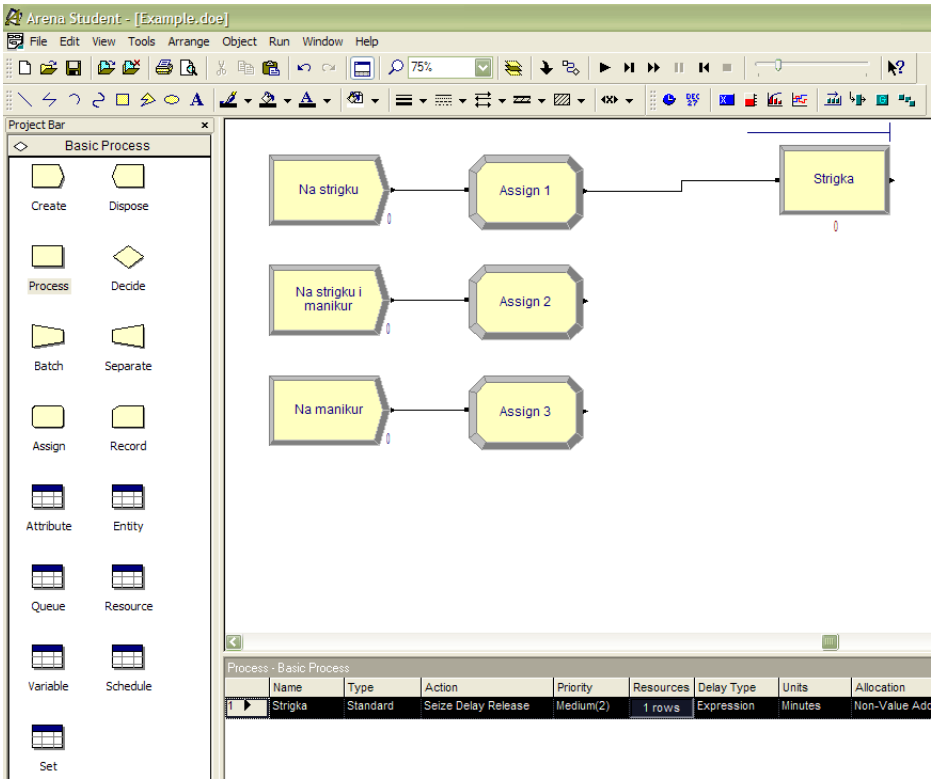


Рис. 1.14. Добавление в модель модуля «Strigka»

Аналогично можно создать процесс «Manikur», отображающий выполнение маникюра. Свойства этого модуля задаются аналогично свойствам процесса «Strigka». Большинство параметров имеют те же самые значения, за исключением Re-

sources и Expression. В поле Resources необходимо указать другое имя ресурса, например, Resources2, т.к. процесс выполняет не ранее введенный ресурс Resources1 (парикмахер), а другой – мастер по маникюру. В поле Expression задается выражение, определяющее значение временной задержки на выполнение маникюра. Например, если на маникюр уходит 30±10 мин., то вводим: UNIF (20, 40).

Соединим модуль «Manikur» с модулем «Assign3», потому что клиенты третьего типа направляются к мастеру по маникюру. Для клиентов второго типа, которые должны посетить и парикмахера, и мастера по маникюру, необходимо разветвить ход выполнения бизнес-процесса. Для этого используется модуль Decide.

8. Использование модуля Decide

Модуль Decide позволяет проверить определенные условия и в зависимости от результата проверки направить сущности по той или иной ветке. В нашем примере клиенты второго типа направляются либо на процесс стрижки, если парикмахер свободен, либо, в противном случае, к мастеру по маникюру.

Перетащим модуль Decide из панели модулей в окно модели. Щелкнем по модулю «Decide1» два раза, опишем его параметры (рис. 1.15). В поле Name (имя) вводим «Parikmaher svoboden?». В поле Type (тип принятия решений) выбираем By Condition, т.к. выбор основан не на вероятности, а на результатах проверки конкретно заданного условия. Для параметра If (тип условия, которое будет проверяться) выбираем значение Expression, т.к. условие – определенное выражение. В поле Value записывается выражение условия. Условие, что парикмахер, обозначенный нами ранее как Resource 1, должен быть свободен записывается следующим образом: STATE(Resource 1) == IDLE_RES.

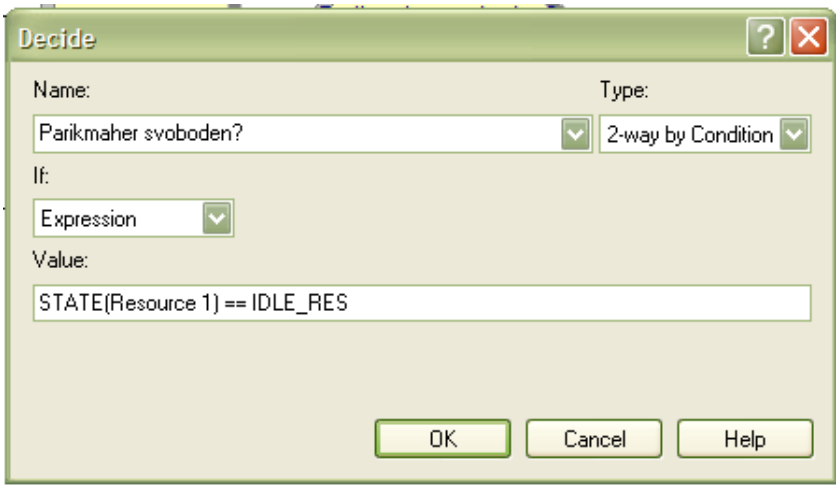


Рис. 1.15. Задание свойств модуля «Decide1»

Теперь нужно соединить этот модуль с другими модулями. У модуля «Parikmaher svoboden?» будет один вход и два выхода – True и False. Через ветку True (истина) проходят сущности в случае, если условие, записанное в выражении Expression, выполняется. Иначе сущности покидают модуль через ветку False (ложь). Соединим модуль «Parikmaher svoboden?» так, чтобы вход поступал от модуля «Assign2», выход True шел на модуль «Strigka», а выход False – на модуль «Manikur» (рис. 1.16).

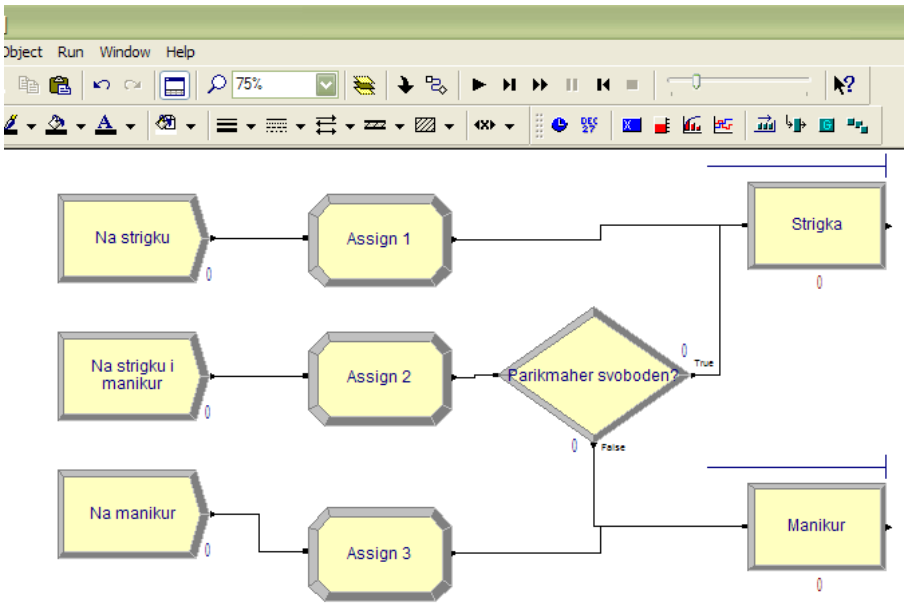


Рис. 1.16. Соединение модуля Decide с другими модулями

В нашем примере недостаточно использовать только один модуль Decide, т.к. после выполнения процесса «Strigka» у клиента второго типа остается еще потребность в выполнении процесса «Manikur», если сначала он был направлен на стрижку. Если же клиент этого типа сначала был направлен к мастеру по маникюру, то после выполнения процесса «Manikur» он должен быть направлен на выполнение процесса «Strigka». Таким образом, и после выполнения процесса «Strigka», и после выполнения процесса «Manikur» нужно изменить значение атрибутов, характеризующих потребность (атрибута Attribute 1, отвечающего за желание клиента подстричься, или атрибута Attribute 2, отвечающего за желание сделать маникюр), а также выполнить проверки, какая потребность у клиента еще остается.

Добавим два модуля типа Assign, соединим модули «Strigka» и «Manikur» с модулями «Assign4» и «Assign5» соответственно (рис. 1.17).

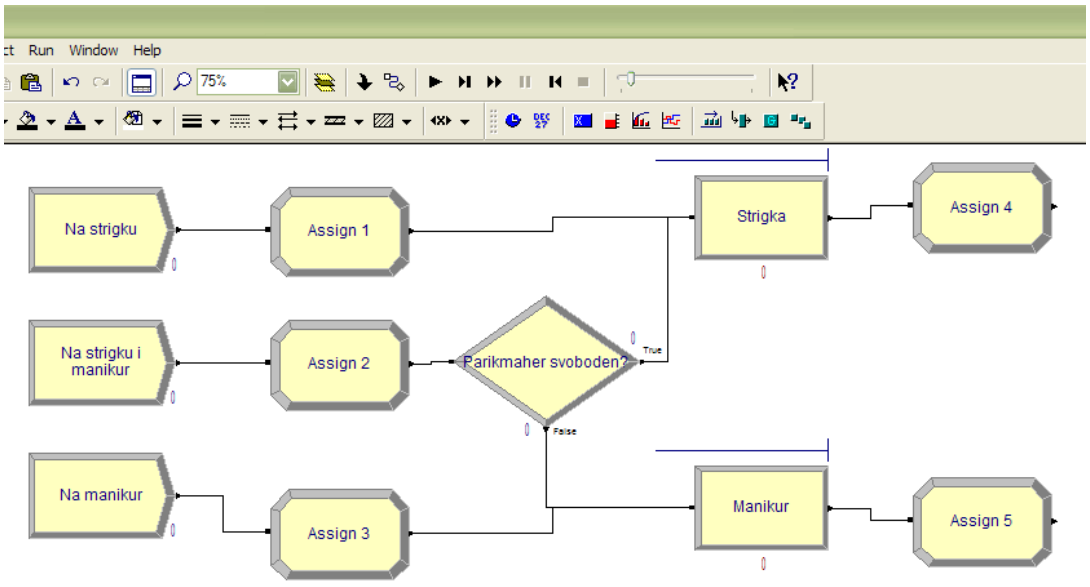


Рис. 1.17. Добавление в модель модулей Assign

Свойства модуля «Assign4» приведены на рис. 1.18. Так как этот модуль, выполняемый после модуля «Strigka», должен отменить потребность в парикмахере, то нажав кнопку «Add…», выбираем Attribute 1 и присваиваем ему значение (New Value) = 0. Аналогично в свойствах модуля «Assign5», выполняемого после модуля «Manikur», устанавливаем значение Attribute 2=0, т.к. потребность в маникюре уже удовлетворена.

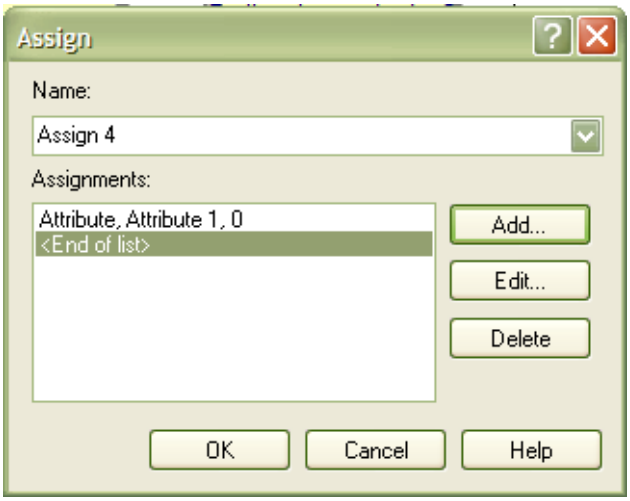


Рис. 1.18. Задание свойств модуля «Assign4»

Добавляем в модель два модуля «Decide2» и «Decide3». Модуль «Decide2», выполняемый после «Assign4», т.е. после отмены потребности в стрижке, будет проверять, осталась ли еще потребность в маникюре (Attribute 2=0?). Модуль «Decide3», выполняемый после «Assign5», т.е. после отмены потребности в маникюре, будет проверять, осталась ли еще потребность в стрижке (Attribute 1=0?).

Свойства модуля «Decide2» приведены на рис. 1.19. В поле Name введено новое имя модуля «Attribute 2 is 0?». В поле Type выбрано значение By Condition, т.к. проверяется выполнение конкретно заданного условия. В поле If выбрано Attribute, т.к. условие – значение атрибута. В поле Named указан атрибут Attribute 2, в поле Is - «==» (проверка на равенство), в поле Value – 0 (сравнение с 0).

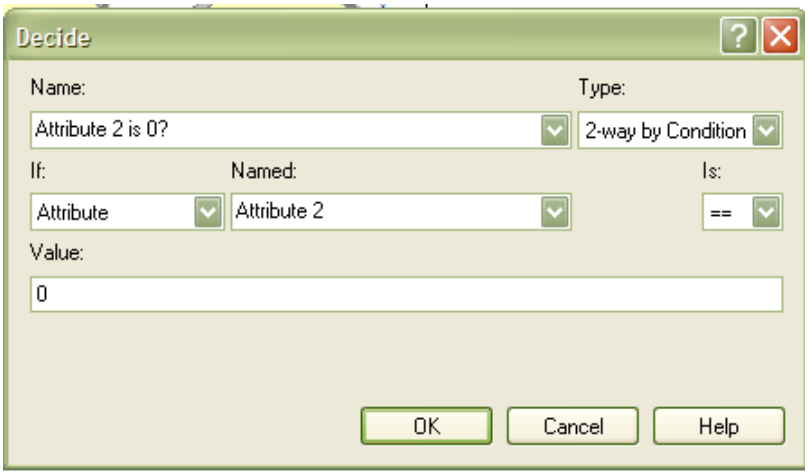


Рис. 1.19. Задание свойств модуля «Decide 2»

Аналогично задаются свойства модуля «Decide3», только в поле Named указывается атрибут Attribute 1, значение которого будет сравниваться с 0.

9. Использование модуля Dispose

Теперь поместим на диаграмму модуль Dispose, предназначенный для завершения потока работ и удаления сущностей (клиентов) из системы. В свойствах модуля изменим значение параметра Name (имя) на «Dovolnii klient» (рис. 1.20).

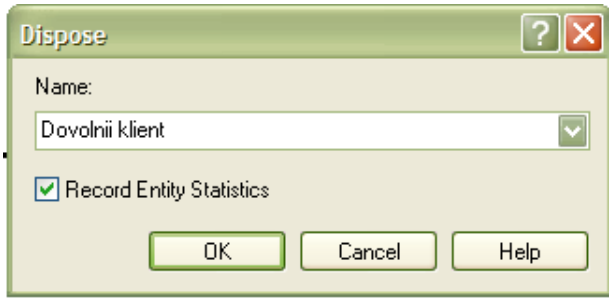


Рис. 1.20. Задание свойств модуля Dispose

Осталось соединить модули. Выход True модуля «Attribute 2 is 0?» типа Decide, выполняемого после стрижки, соединяем с модулем «Dovolnii klient» типа Dispose, т.к. в этом случае у клиента уже нет потребностей и он покидает салон. Выход False этого модуля соединяем с модулем «Manikur», т.к. у клиента осталась потребность сделать маникюр. Аналогично выход True модуля «Attribute 1 is 0?» типа Decide, выполняемого после маникюра, соединяем с модулем «Dovolnii klient» типа Dispose, а выход False – с модулем «Strigka». В результате получаем модель деятельности салона красоты (рис. 1.21).

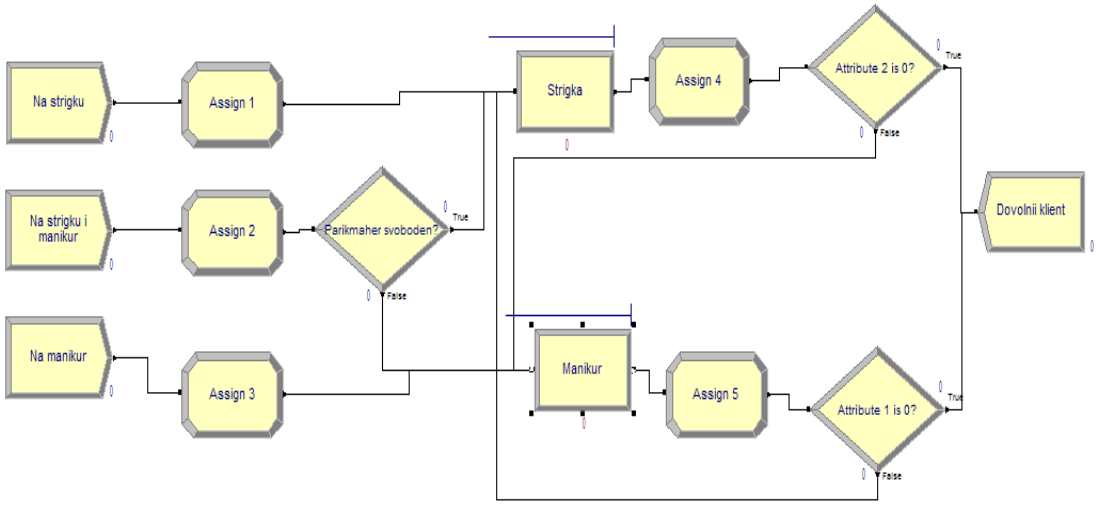


Рис. 1.21. Законченная модель деятельности салона красоты

10. Доработка модели по индивидуальному заданию

Завершите создание имитационной модели для бизнес-процесса, выбранного вами на шаге 1 в качестве индивидуального задания. Подробное описание ранее рассмотренных модулей, а также других приведено в приложении 2.

Лабораторная работа №2
«Проигрывание имитационной модели Arena»

Цель работы: Научиться воспроизводить процесс функционирования системы во времени с помощью инструментального средства Arena. Провести эксперименты с моделью с разными входными данными. Ознакомиться с основными объектами визуализации программного пакета Arena.

Порядок выполнения работы

1. Подготовка к запуску

Чтобы подготовить модель к прогонам, необходимо ввести общую информацию о проекте и задать продолжительность прогона.

Откроем окно свойств проекта, используя меню Run → Setup → Project Parameters. Вид окна приведен на рис. 2.1. В поле Project Title введено имя проекта – Example, в поле Analyst Name указано имя разработчика проекта. В области Statistics Collection отмечаются необходимые параметры для сбора статистики. Нас интересует статистика по стоимости, по сущностям (клиентам), очередям, процессам и ресурсам (рис. 2.1).

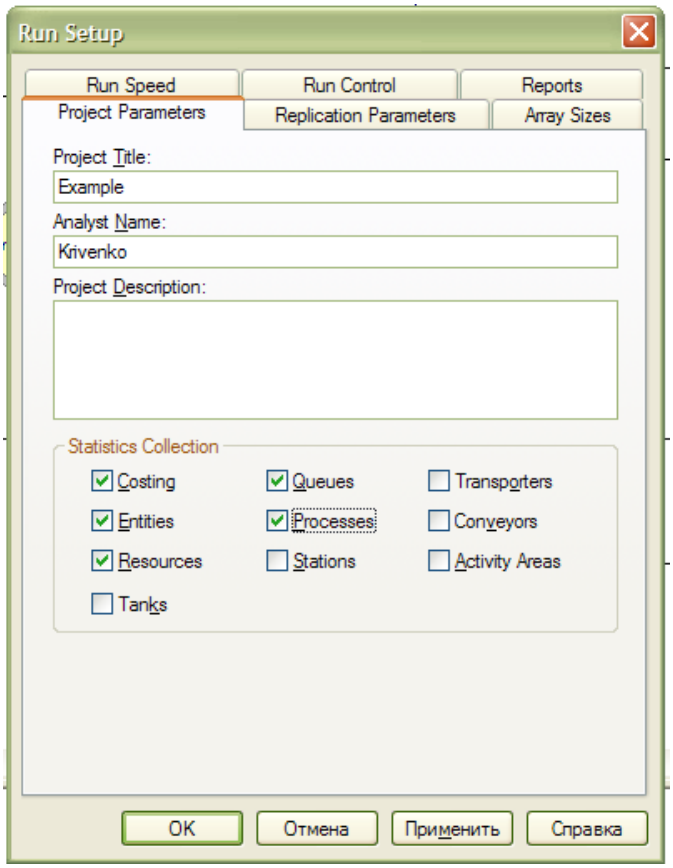


Рис. 2.1. Задание информации о проекте

Чтобы задать параметры прогона, в том же окне выберем вкладку Replication Parameters. Допустим, мы хотим смоделировать восьмичасовой рабочий день салона красоты. В поле Replication Length введем продолжительность прогона – 8, в поле Time Units выберем единицу измерения продолжительности – часы (рис. 2.2). Нажмем OK.

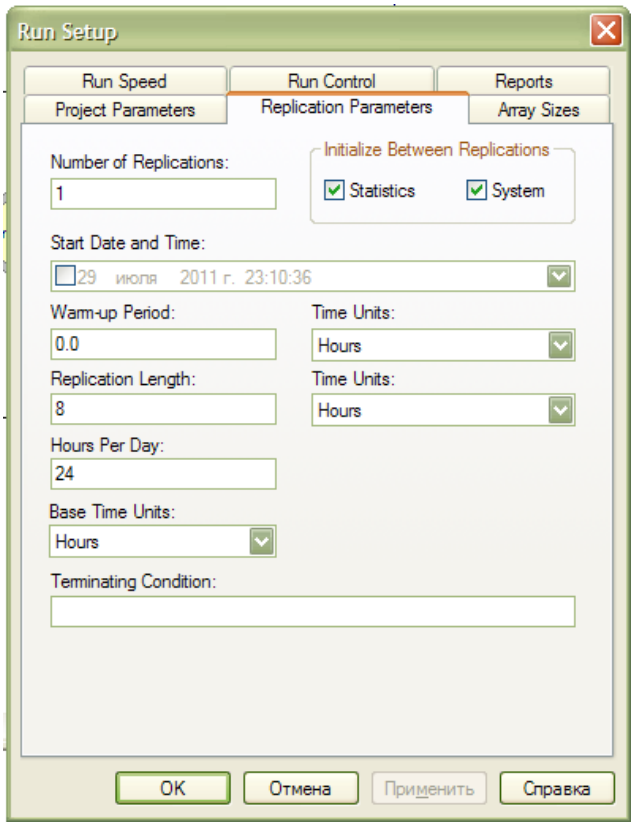


Рис. 2.2. Задание параметров прогона

На вкладке Run Speed можно задать скорость прогона, задав значение параметру Animation Speed (рис. 2.5). Однако задавать скорость удобнее во время прогона с помощью специальной шкалы, использование которой будет описано дальше.

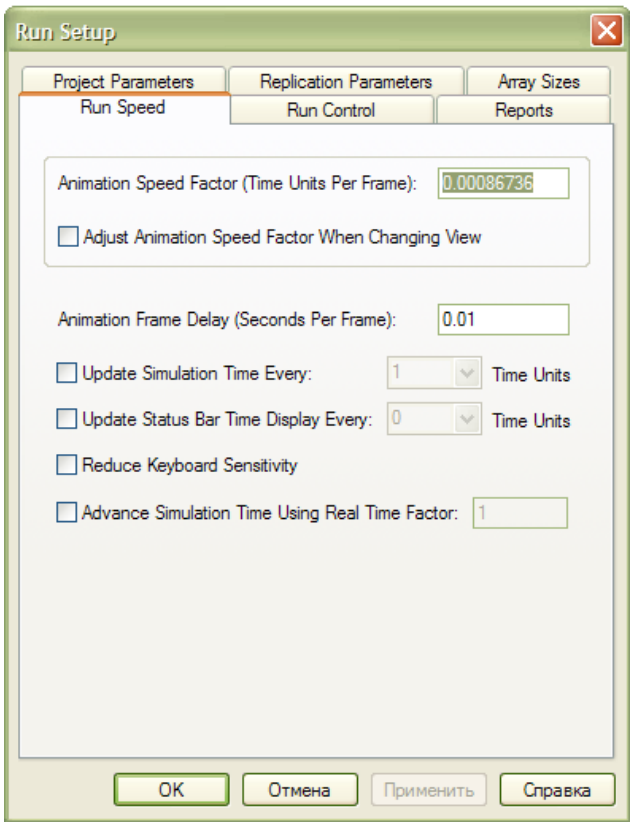


Рис. 2.3. Задание скорости прогона

2. Осуществление прогона

Теперь модель содержит всю необходимую информацию для осуществления запуска. Чтобы начать прогон, щелкнем кнопку Go на панели инструментов (рис. 2.4), или из меню Run → Go.

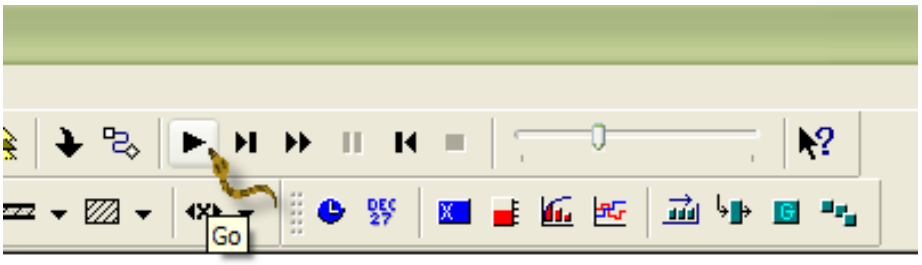


Рис. 2.4. Инструменты управления прогоном

Во время прогона пользуемся шкалой скорости прогона, расположенной на панели инструментов рядом с другими кнопками управления имитацией (рис. 2.4). Для уменьшения скорости сдвигаем указатель влево, для увеличения – вправо.

В любой момент можно приостановить прогон. Для этого нужно нажать на кнопку Pause  панели инструментов (рис. 4.4) или клавишу Esc.

В процессе прогона нам будут доступны результаты имитационного моделирования, например, количество сущностей, созданных модулями Create, или количество сущностей, направленных к тому или иному процессу, или количество сущностей, обработанных тем или иным процессом к текущему моменту времени (рис. 2.5).

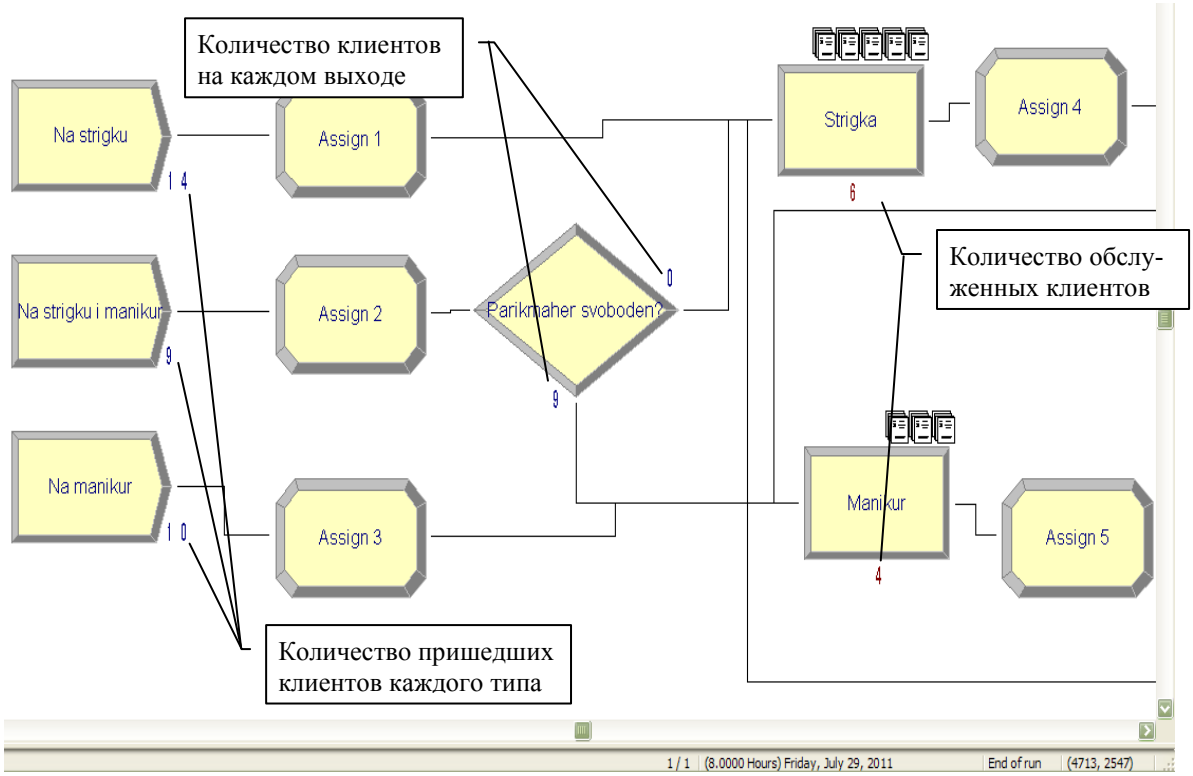


Рис. 2.5. Просмотр хода моделирования

3. Просмотр отчетов моделирования

По окончании установленного времени прогона можно просмотреть отчеты по статистике, собранной за время работы модели. Сразу после прогона автоматически появится сообщение, хотите ли вы посмотреть отчеты (рис. 2.6). Нажмите «Да». По умолчанию откроется окно отчета.

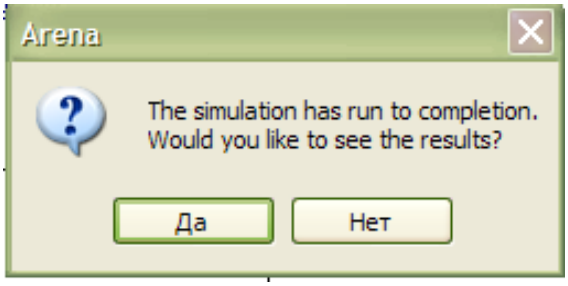


Рис. 2.6. Запрос просмотра отчета

Слева в окне окно отчета Category Overview (рис. 2.7) находится дерево всех показателей, доступных для просмотра. Справа выводится отчет, содержащий собранные во время эксперимента значения показателей. Например, в отчете по сущностям отражается время (среднее, минимальное, максимальное) обработки сущности в том или ином процессе, время ожидания обработки, время нахождения в системе (см. рис. 2.7).

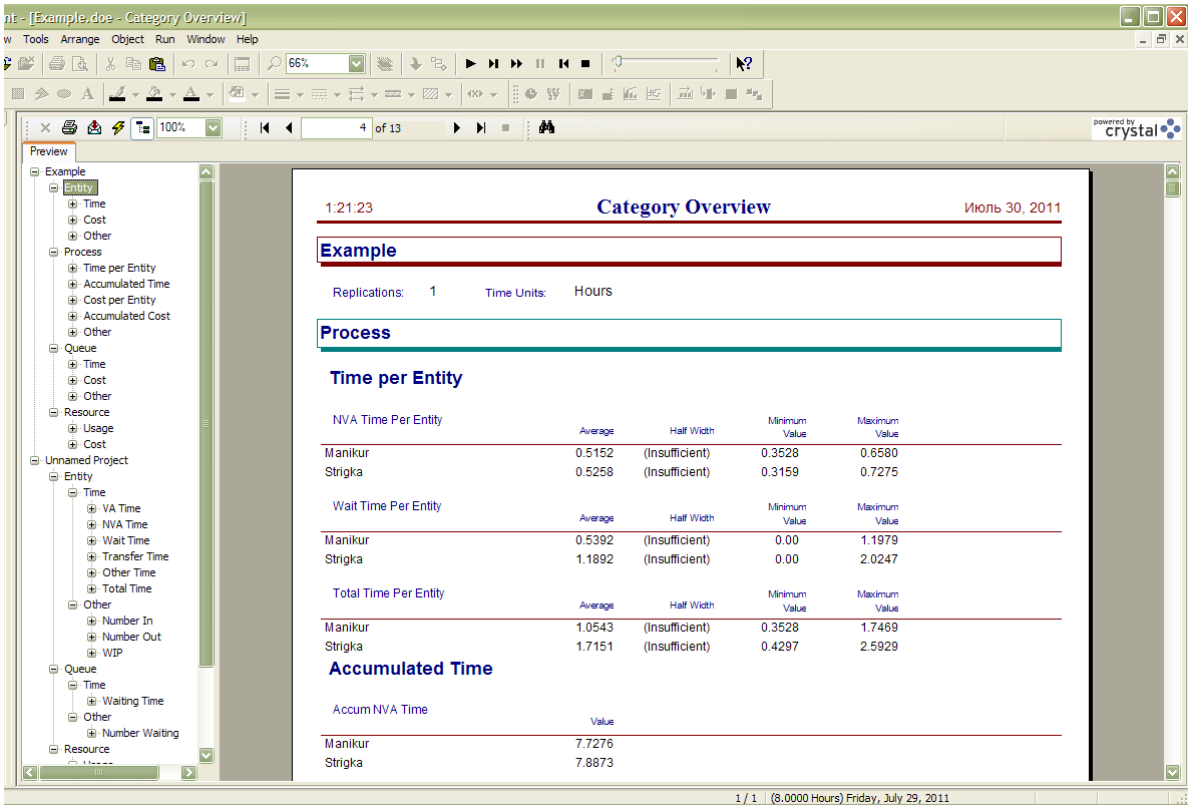


Рис. 2.7. Отчет по результатам моделирования

Чтобы завершить прогон и вернуться к редактированию модели, нажмите кнопку End ■ на панели инструментов (рис. 2.4).

4. Вывод результатов моделирования на рабочее поле

Для того чтобы можно было просматривать результаты моделирования непосредственно на рабочем поле модели, воспользуемся кнопкой Variable на панели инструментов (рис. 2.8).

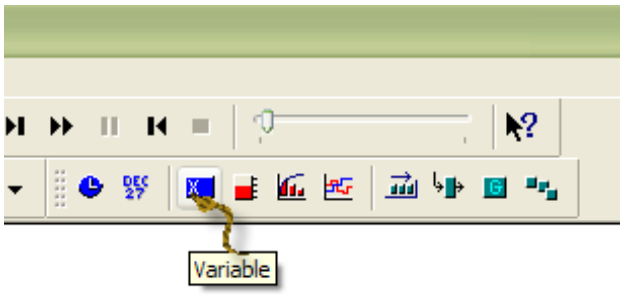


Рис. 2.8. Вызов окна описания переменной

В появившемся окне (рис. 2.9) опишем параметры той переменной, значения которой мы хотим просматривать в рабочем поле. Допустим это переменная, отражающая количество клиентов, пришедших на стрижку.

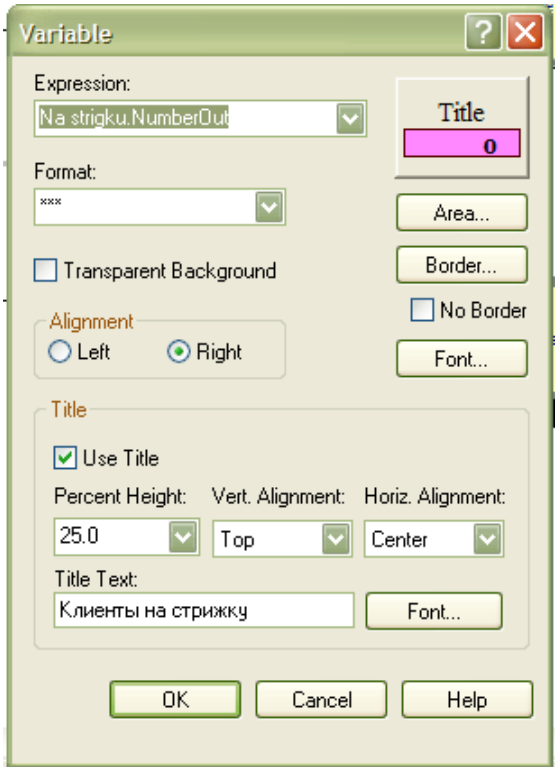


Рис. 2.9. Задание параметров переменной

В поле Expression введем переменную – «Na strigku.NumberOut», т.к. это параметр NumberOut модуля «Na strigku», создающего клиентов на стрижку. В поле Format укажем «***», т.к. значение показателя целое и не превышает трехзначное число. Нажав кнопку Area, выбираем цвет заливки фигуры, на которой будет отображаться значение переменной (Transparent Background – заливка отсутствует). Нажав кнопку Border, можно выбрать цвет границы фигуры (No Border – граница отсутствует). По кнопке Font можно выбрать шрифт. В поле Alignment выбираем выравнивание значения (Left – по левому краю, Right – по правому).

Если требуется заголовок, ставим галочку в поле Use Title. Указываем размер заголовка (Percent Height – 25), вертикальное выравнивание (Vert. Alignment – Top, т.е. сверху), горизонтальное выравнивание (Horiz. Alignment – Center, т.е. по центру), текст заголовка (Title Text - Клиенты на стрижку). По кнопке Font можно вызвать окно для выбора шрифта заголовка.

Теперь выберем место в области построения модели для размещения показателя. Растянем его до нужного размера (рис. 2.10).

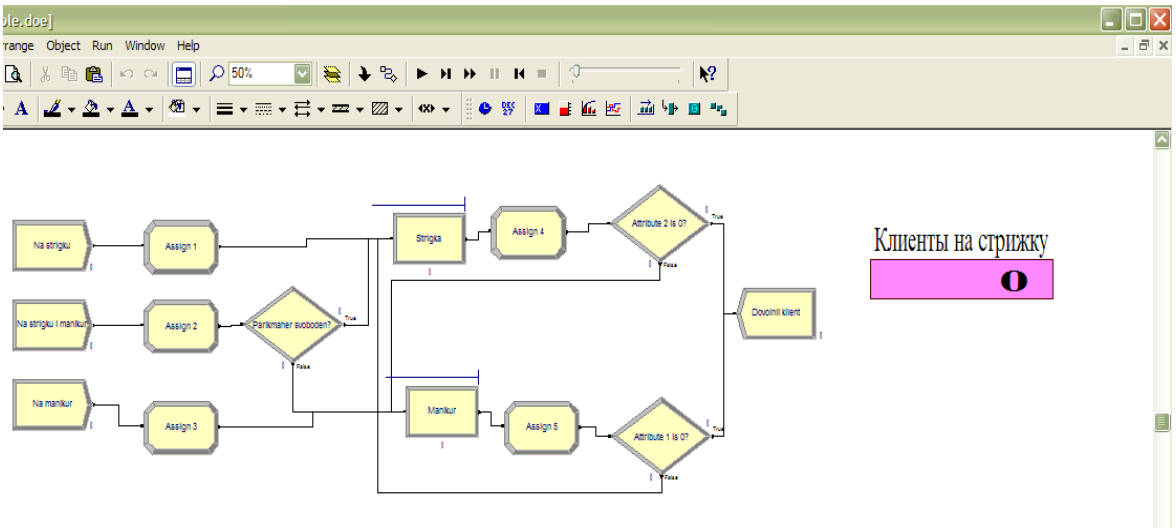


Рис. 2.10. Размещение параметра в рабочем поле

Аналогично можно определить и разместить в рабочем поле переменные: «Na Manikur.NumberOut» – количество клиентов, пришедших сделать маникюр; «Na strigku i manikur.NumberOut» – количество клиентов, желающих сделать и стрижку, и маникюр; «Dovolnii klient.NumberOut» – количество обслуженных в салоне клиентов (параметр NumberOut модуля «Dovolnii klient» типа Dispose).

Рассмотрим, как вывести вычисляемые переменные. Допустим мы хотим разместить переменную, показывающую количество клиентов, не обслуженных парикмахером. В окне для ввода параметров переменной (рис. 2.9) в поле Expression выберем Build Expression. Откроется окно построителя выражений (рис.2.11).

В появившемся окне строим выражение следующим образом. В секции Expression Type (тип выражения) выбираем: Process → Number In Process. В поле Process Name вводим имя процесса, для которого строится выражение – Strigka. В секции записи выражения Current Expression вводим выражение «Strigka.WIP»/

Аналогично можно задать параметры для переменной, показывающей количество клиентов, не обслуженных мастером по маникюру, только в поле Process Name построителя выражений нужно задать имя «Manikur».

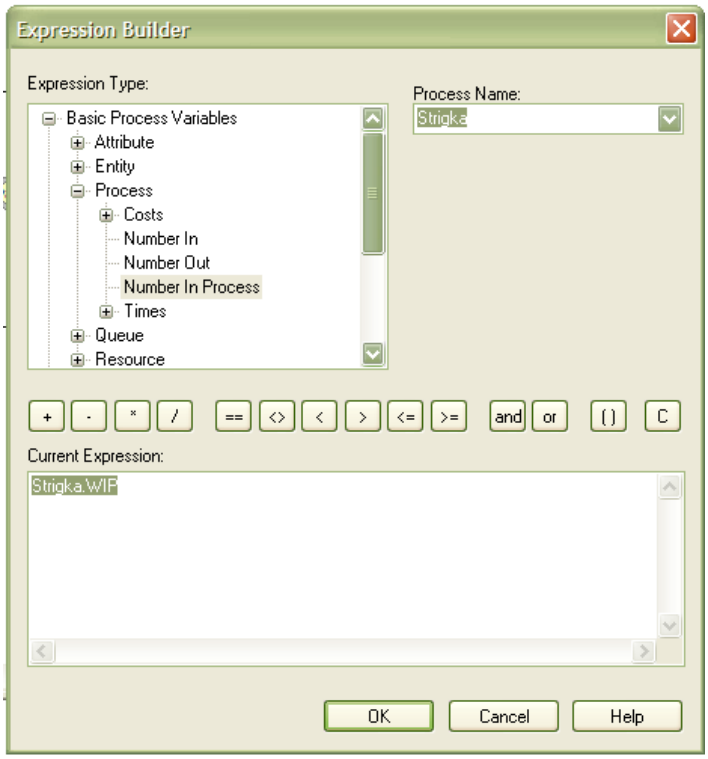


Рис. 2.11. Построение выражения для переменной

После того, как все переменные будут определены и размещены, можно будет просматривать результаты прогона (полученные значения переменных) непосредственно в поле модели (рис. 2.12).

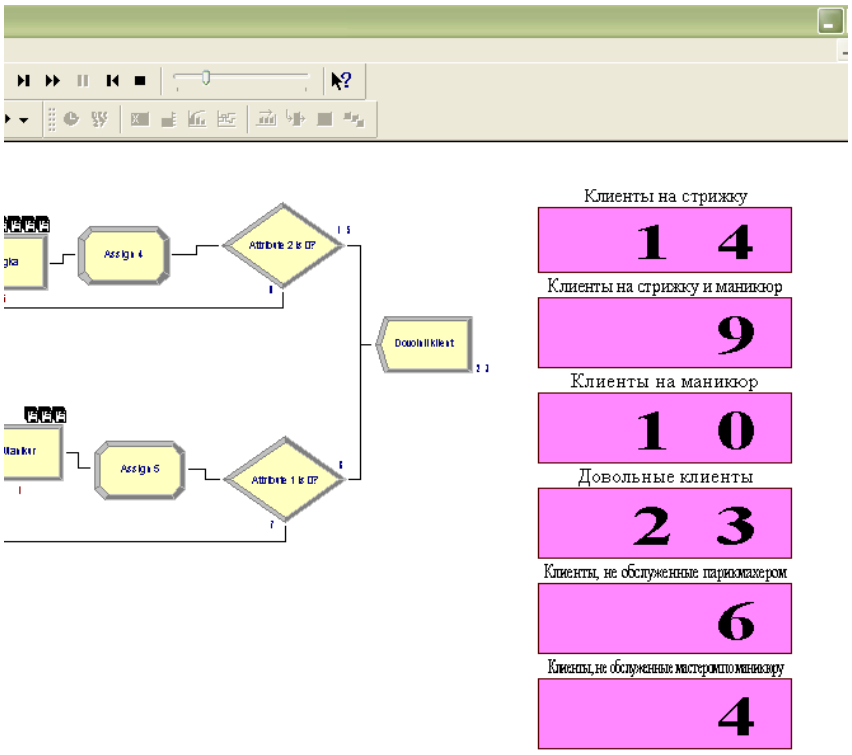


Рис. 2.12. Просмотр результатов прогона в рабочем поле

5. Эксперименты с моделью

Эксперименты на модели позволяют проверить различные гипотезы, как изменятся результаты работы системы при изменении тех или иных параметров. Можно изменить логику процесса, изменить время поступления сущностей в модулях Create или время обработки сущностей в модулях Prosess, можно добавить новые модули или новые типы сущностей и др.

Например, мы можем проверить, как изменятся результаты работы салона красоты, если парикмахер и мастер по маникюру после повышения квалификации сократят время обслуживания клиента (на стрижку будет уходить не 30±15 мин., а всего лишь 18±15 мин., а на маникюр не 30±10 мин., а 25±10 мин.).

Проведем эксперимент с новыми данными. Изменим значения параметров модулей «Strigka» и «Manikur». В окне свойств модуля «Strigka» (см. рис. 3.13) в поле Expression введем выражение: UNIF (3, 33). Аналогично для модуля «Manikur» введем выражение UNIF (15, 35).

Сделаем прогон модели с новыми параметрами и получим новые значения переменных, выведенных на рабочее поле (рис. 2.13).

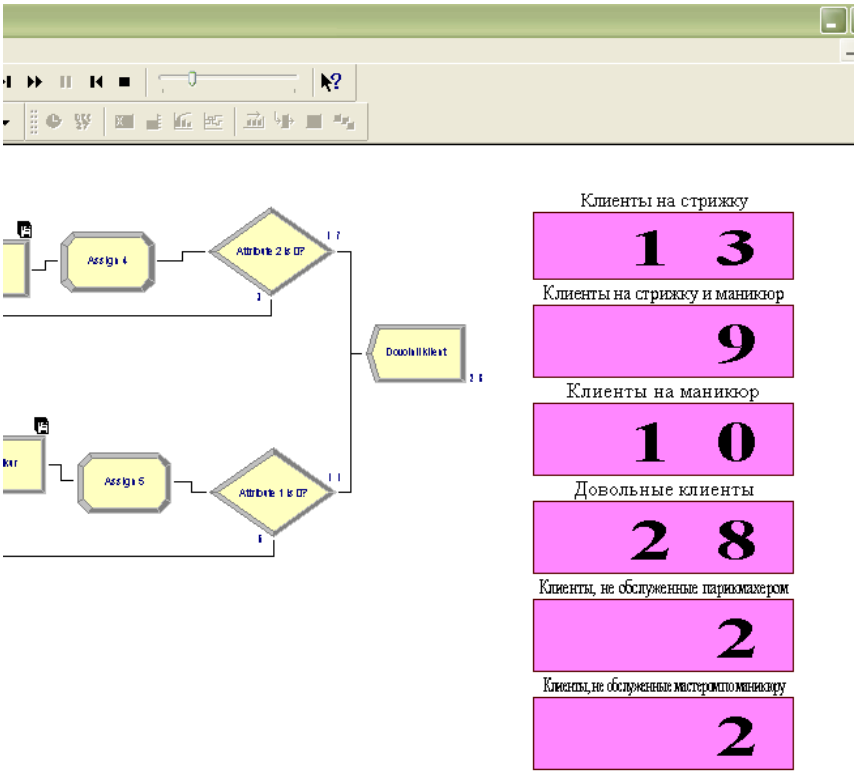


Рис. 2.13. Результаты прогона после изменения параметров модели

Сравните результаты прогона с предыдущими результатами (до изменения параметров) и сделайте выводы.

Сравнивая значения показателей на рис. 2.12 и 2.13, видим, что повышение квалификации благоприятно сказались на количестве обслуженных клиентов. Количество не обслуженных клиентов у мастера по маникюру сократилось вдвое, а у парикмахера – втрое. Можем сделать вывод, что салону красоты стоит отправить своих сотрудников на курсы повышения квалификации.

6. Добавление текста и часов в поле модели

Добавление визуальных элементов в поле модели, например, поясняющего текста (текстовых блоков), часов, показывающих модельное время, и т.д. может повысить наглядность модели и хода имитации.

Добавим в нашу модель две надписи: «Процесс работы салона красоты» (название модели) и «Показатели» (название столбца, где находятся необходимые показатели). Для этого щелкнем кнопку Text на панели инструментов (рис. 2.14).

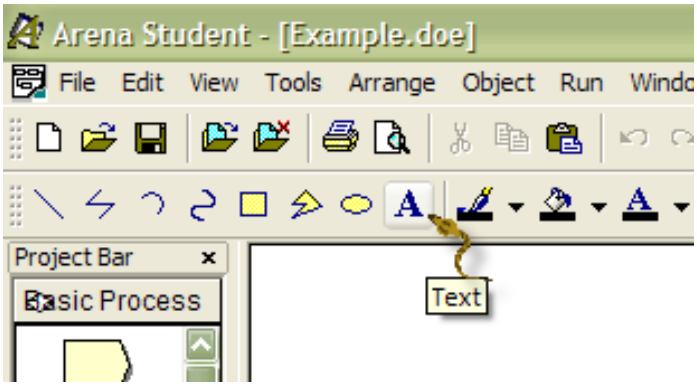


Рис. 2.14. Выбор инструмента для добавления текста

В появившемся окне введем нужный текст (например, «Процесс работы салона красоты»). Нажав «OK», выберем место в области построения модели для размещения текста. Изменим цвет текста кнопкой Text Color  (рис. 2.14) на более яркий. Подобным образом создадим вторую надпись (рис. 2.15).

Рис. 2.15. Добавление текстовых блоков в поле модели

Часы необходимы для того, чтобы наглядно видеть, сколько рабочего времени прошло. Щелкнем кнопку Clock на панели инструментов (рис. 2.16).

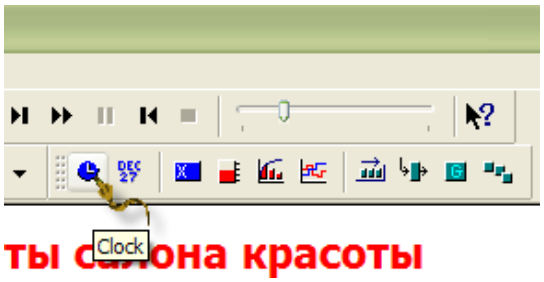


Рис. 2.16. Выбор инструмента для добавления часов

В появившемся окне опишем параметры, необходимые для выведения на экран электронных часов (рис. 2.17). Пусть наш салон красоты работает с 10 до 18 часов. Тогда в секции Stating Time (начальное время) указываем 10 часов 0 мин. 0 сек. В секции Display выбираем Digital – изображение электронных часов. Формат электронных часов (Time Format) – 24 Hour. Нажав на кнопку Area, выбираем цвет часов (Transparent Background – цвет отсутствует). С помощью кнопки Border вызываем окно для выбора цвета рамки (No Border – рамка отсутствует). Кнопка Digits позволяет выбрать цвет цифр.

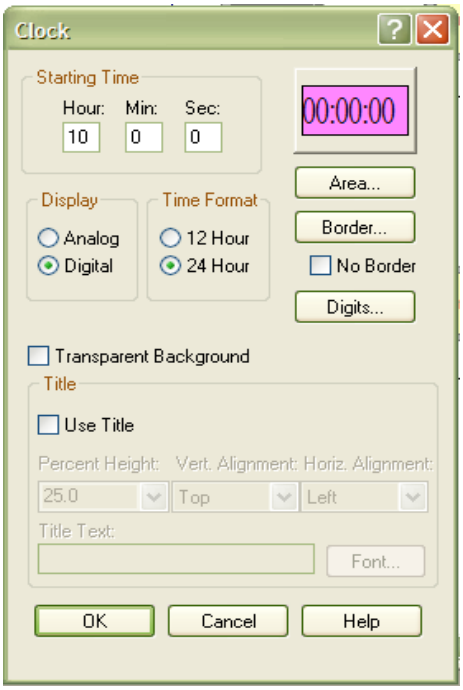


Рис. 2.17. Задание параметров часов

Нажмем «OK». Выберем место в области построения модели для размещения часов. Растянем их до нужного размера (рис. 2.18).

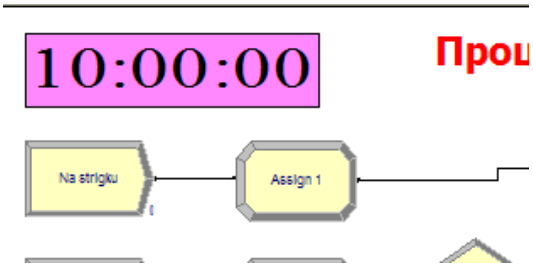


Рис. 2.18. Добавление часов

Запустив модель, видим, как изменяется время на часах.

7. Визуализация состояний ресурсов

Изображения, связанные с ресурсами, повышают наглядность имитации. Например, мы хотим отобразить с помощью различных картинок различные состояния (занят или свободен) парикмахера и мастера по маникюру.

Чтобы изобразить парикмахера, выполним следующие шаги. Нажмем кнопку Resource  на панели инструментов (ее изображение есть на рис. 2.16). В появившемся окне выберем из выпадающего списка Identifier тот ресурс, который будем анимировать. В нашем примере – Resource1 (парикмахер). Нажав кнопку Open, мы можем выбрать ту библиотеку изображений, которая соответствует нашему ресурсу. В рассматриваемом примере это библиотека People.plb (рис. 2.19).

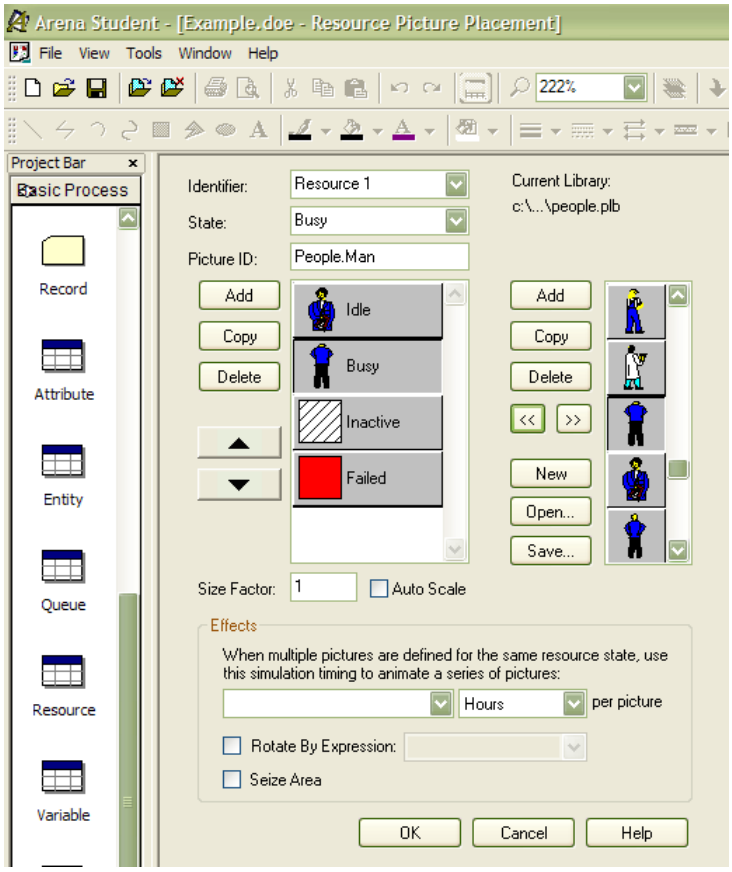


Рис. 4.19. Присвоение изображений ресурсам

Чтобы присвоить изображение состоянию «свободен» выбранного ресурса (парикмахера), необходимо в левой колонке выбрать кнопку Idle (свободен) и в правой колонке выбрать нужную картинку. После нажатия кнопки “<<” выбранная картинка будет присвоена состоянию свободного ресурса. Аналогично можно присвоить изображение состоянию «занят» выбранного ресурса: в левой колонке выбрать кнопку Busy (занят), в правой колонке выбрать нужную картинку и нажать кнопку “<<”.

После того, как определены изображения для каждого состояния ресурса, нажмем OK и выберем место в области построения модели для размещения изображения. Растянем изображение до нужного размера. Добавим надпись «Парикмахер».

Точно таким же образом и для других ресурсов можно присвоить изображения (для обоих состояний) и разместить их в поле модели.

На рис. 2.20 можно увидеть, как будет выглядеть модель работы салона красоты после добавления изображений парикмахера и мастера по маникюру.

Рис. 2.20. Добавление изображений ресурсов в поле модели

8. Добавление графиков

Графики – это еще один визуальный элемент, позволяющий наглядно отобразить результаты имитации. Допустим, мы хотим разместить в поле модели график, показывающий количество клиентов, находящихся у мастеров (у парикмахера и мастера по маникюру) в процессе обслуживания и ожидания в различные моменты времени.

Нажмем кнопку Plot на панели инструментов (ее изображение есть на рис. 4.16). Чтобы добавить выражение, отображаемое на графике, нажмем Add.

В появившемся окне Plot Expression (рис. 2.21) зададим наименование графика (Name – «Клиенты у мастеров») и требуемое выражение (Expression).

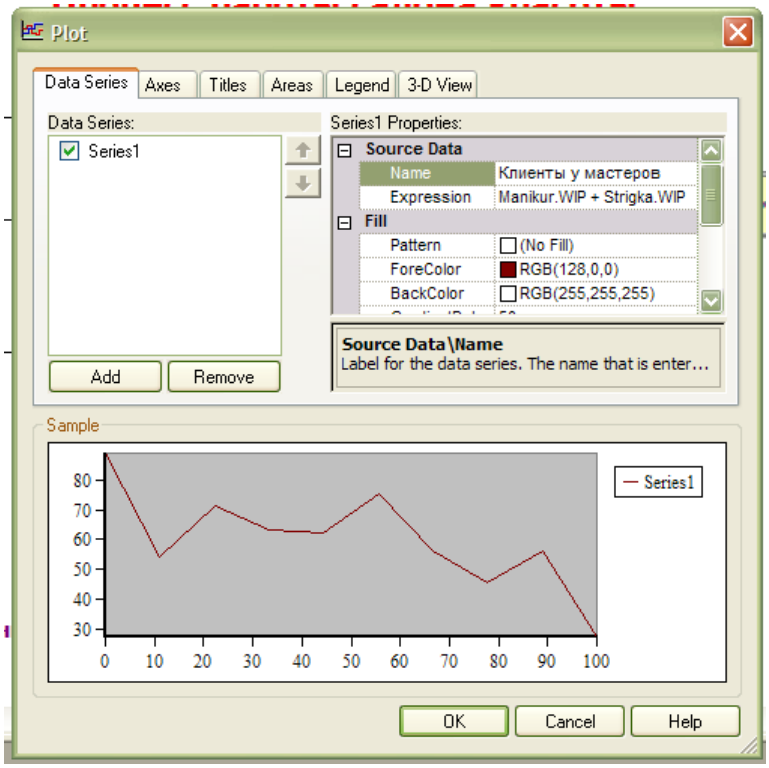


Рис. 2.21. Задание параметров графика

Выражение можно выбрать из предлагаемого выпадающего списка (если требуемое выражение есть в списке), или нажать правой кнопкой в поле Expression и открыть построитель выражений. С помощью построителя выражений мы создадим следующее выражение: «Manikur.WIP + Strigka.WIP», т.е. график будет выводить количество клиентов, находящихся у мастеров в данный момент (рис. 2.21).

Во вкладке Axes изменим значение параметра Maximum на 20 (рис. 2.22). Нажмем ОК.

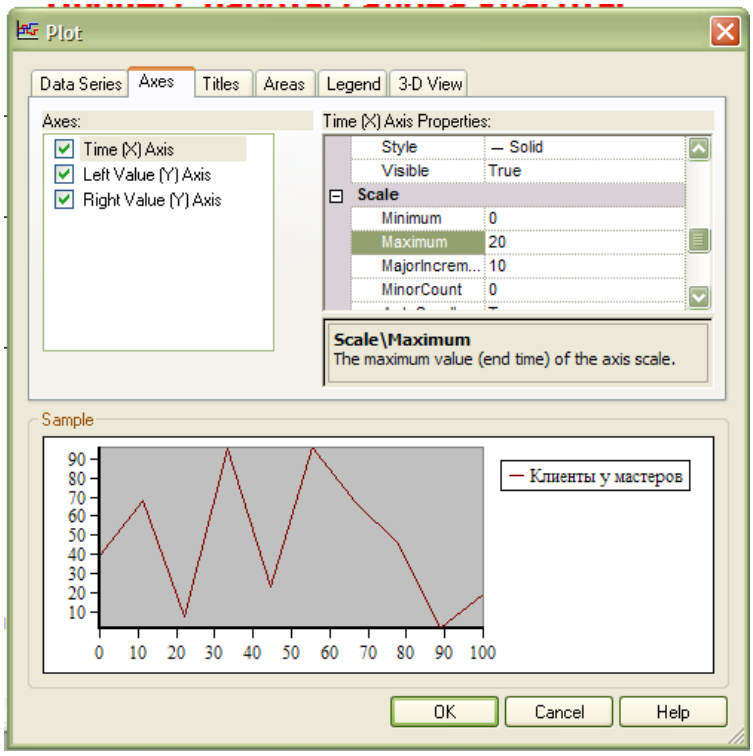


Рис. 2.22. Задание максимального количества повторений

Укажем место в области построения модели для размещения графика, растянем его до нужного размера (рис. 2.23).

Рис. 2.23. Размещение графика в поле модели

9. Изменение изображений сущностей

В процессе имитации если в модуль процесса поступает сущность, ее изображение появляется над модулем процесса. Стандартное изображение сущности – в виде стопки бумаг ( ). Но в нашей модели сущности – это люди. Можно изменить стандартное изображение, например, на изображение женщины, т.к. в основном клиенты салона красоты – это женщины.

Щелкните по модулю данных Entity на панели основных процессов (Basic Process Panel). В окне свойств модуля щелкнем два раза по ячейке с номером 1 (рис. 2.24).

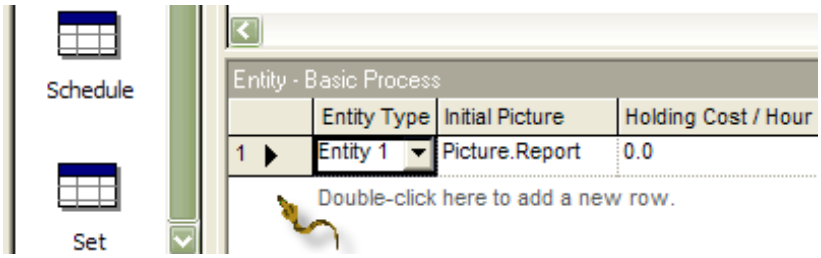


Рис. 2.24. Выбор сущности для смены изображения

В появившемся окне параметр Initial Picture отвечает за изображение (рис. 2.25). Зададим значение этого параметра – Picture.Woman ( ).

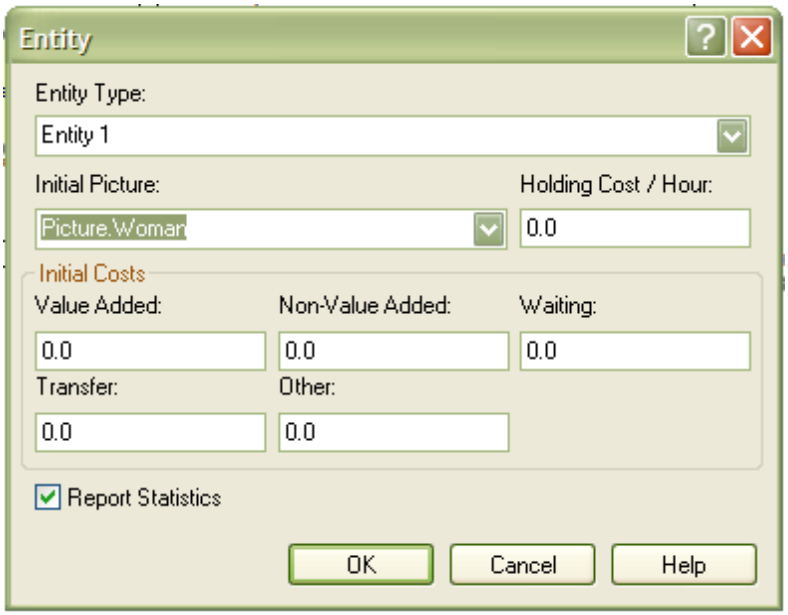


Рис. 2.25. Задание изображения для сущности

Теперь во время прогона модели можно наблюдать за работой мастеров и динамикой изменения количества клиентов, находящихся в процессе (рис. 2.26).

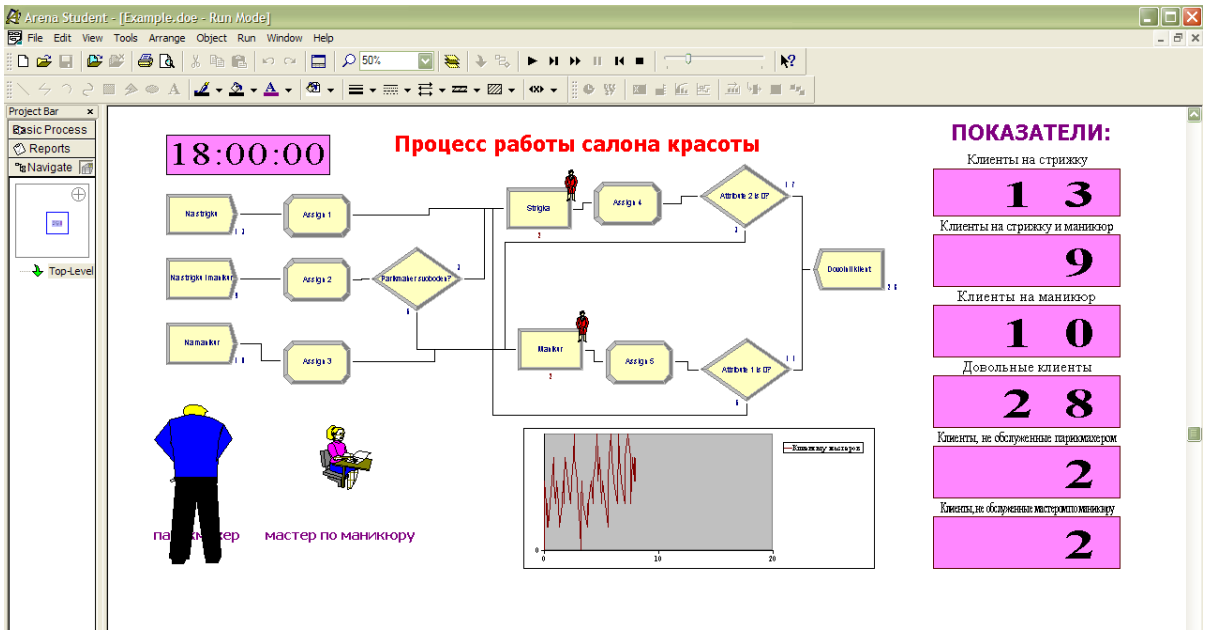


Рис. 2.26. Окончательный вид модели

10.Выполнение изменений модели

Выполните все изменения модели, предусмотренные индивидуальным заданием (см. приложение 1). Выполните проигрывание обновленной модели. Сделайте выводы.

Литература

1. Силич В.А., Силич М.П. Моделирование и анализ бизнес-процессов : учебное пособие. - Томск : Томск. гос. ун-т систем управления и радиоэлектроники, 2011. – 212 с.
2. Замятина О. М. Компьютерное моделирование. Учебное пособие. – Томск: Изд-во ТПУ, 2007. – 115 с.
3. Интернет-магазин и портал INTERFACE.RU – Разнообразные программные продукты и их описание [Электронный ресурс]. Режим доступа к сайту: www.interface.ru.
4. Портал SWSYS – электронная версия журнала «Программные продукты и системы» [Электронный ресурс]. Режим доступа к сайту: swsys.ru.

Приложение 1

Варианты индивидуальных заданий

1. Работа парикмахерской

В парикмахерскую могут приходить клиенты двух типов. Клиенты первого типа желают только стричься. Распределение интервалов их прихода 35+10 мин. Клиенты второго типа желают постричься и побриться. Распределение интервалов их прихода 60±20 мин. Парикмахер обслуживает клиентов в порядке «первым пришел – первым обслужен». На стрижку уходит 18±6 мин., а на бритье 10+2 мин.

Доходы от работы парикмахерской определяются количеством клиентов, обслуженных в течение рабочего дня (стоимость стрижки – 100 рублей, бритья – 20 рублей), убытки определяются временем простоев парикмахера (в отсутствие клиентов) и количеством необслуженных клиентов в очереди.

Моделирование проведите для рабочей недели (6 дней по 8 часов).

После разработки модели внесите в нее следующие дополнения и изменения:

1. Клиенты первого типа имеют анимационную картинку «Woman» (в виде женщины), а клиенты второго типа – «Man».

2. Задайте анимацию ресурсу «Парикмахер» в состоянии «свободен» (Idle) в виде изображения на рис. П1.1, *а*, и в состоянии «занят» (Busy) – в виде рис. П1.1, *б*.

3. Измените правило обслуживания: приоритет в обслуживании имеют женщины (клиенты первого типа).

4. Рассмотрите возможность ввода в модель второго парикмахера. Как измениться доход парикмахерской?

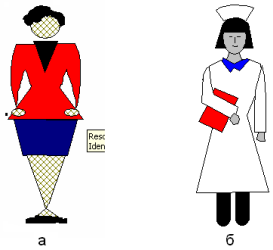


Рис. П1.1. Анимационная картинка ресурса «Парикмахер»:
а - ресурс свободен; *б* - ресурс занят

2. Работа сборочного цеха

В сборочный цех поступают детали трех видов. Детали первого типа (Д1) поступают 20±3 мин (наиболее часто 20 мин). Детали второго типа (Д2) – 16±5 мин. Детали третьего типа (Д3) – 20 мин. Как только сборщику поступают три детали (любые), он производит монтаж готового изделия за 5 мин. Из собранных изделий 15 % бракованные. Если изделие бракуется в первый раз, то оно поступает на повторный монтаж к сборщику. Если изделия бракуются 2 раза, то они идут в отходы (10 мин). Не бракованные изделия упаковываются по 5 штук за 3 минуты упаковщиком.

Смоделируйте 8-часовой рабочий день.

Выполните следующие задания:

1. Определите каждому типу деталей свою анимационную картинку.

2. Определите анимационную картинку готовому изделию и упакованному изделию.

3. Задайте анимационную картинку ресурсам «Сборщик» и «Упаковщик», когда они свободны и заняты.

4. Соберите статистику по бракованным изделиям (отходы и один раз бракованные), количеству упаковок, по загруженности ресурсов «Сборщик» и «Упаковщик».

5. Измените модель следующим образом: сборщик собирает изделие из деталей разного типа, и готовые не бракованные изделия складируются. Один раз в 10 часов из гаража выезжает грузовик и забирает со склада все упаковки.

3. Работа системы сбора информации

Распределенный банк данных системы сбора информации организован на базе ЭВМ, соединенных дуплексным каналом связи. Поступающий запрос обрабатывается на первой ЭВМ, и с вероятностью 50 % необходимая информация обнаруживается на месте. В противном случае необходима посылка запроса во вторую ЭВМ.

Запросы поступают через 10 ± 3 с, первичная обработка запроса занимает 2 с, выдача ответа требует 18 ± 2 с, передача по каналу связи занимает 3 с. Временные характеристики второй ЭВМ аналогичны первой.

Смоделируйте прохождение 400 запросов.

Определите необходимую емкость накопителей перед ЭВМ, обеспечивающую безотказную работу системы, и функцию распределения времени обслуживания заявки.

Выполните следующие задания:

1. В систему первоначально поступают сущности в виде дискет, а затем преобразовываются в самолеты и лодки.
2. Первые 200 запросов идут по ветке True, остальные – по False
3. Первые 30 мин. все запросы шли по True, остальные – по False
4. Первые 200 запросов проходили первичную обработку 2 с, остальные – 4 с.
5. Запросы моделируются с разными приоритетами: в модуле условия с большим приоритетом – по True, с меньшим – по False.

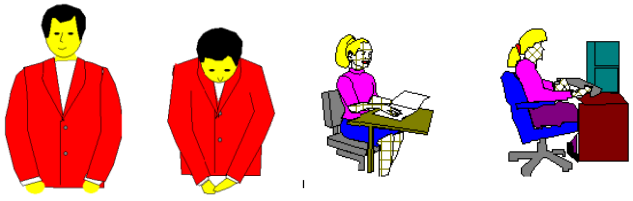
4. Обработка запросов

В компанию поступают запросы (20 ± 4 мин.). Поступающий запрос обрабатывается двумя сотрудниками, причем первый сотрудник обрабатывает 75 % запросов, второй обрабатывает остальные запросы. Первичная обработка запроса занимает 23 минуты, выдача ответа требует 18 ± 5 мин., как у первого, так и у второго сотрудника.

Смоделируйте прохождение 350 запросов.

Выполните следующие задания:

1. Определите количество запросов, обработанных каждым сотрудником за 24 часа.
2. Создайте анимационные картинки ресурсам, когда они свободны и заняты (см. рис. П1.2, *а*, *б*, *в*, *г*).



а *б* *в* *г*

Рис. П1.2. Анимационная картинка ресурсов:
а – «Сотрудник1» свободен; *б* - «Сотрудник1» занят,
в – «Сотрудник2» свободен; *г* - «Сотрудник2» занят

Сделайте следующие изменения модели:

1. В систему первоначально поступают сущности в виде телефонных звонков, а затем к первому сотруднику приходят в виде отчетов, а ко второму сотруднику – в виде дискет.
2. Первые 50 запросов идут к первому сотруднику на обработку, остальные – ко второму.
3. Первые 4 часа все запросы идут ко второму сотруднику на обработку, остальные – к первому.
4. Первые 150 запросов проходят первичную обработку 23 мин., остальные – 30 минут.
5. На обработку поступают 2 вида запросов (телефонные звонки и письма). Причем при первичной обработке у телефонных звонков приоритет выше, чем у писем.

5. Работа слот-бара

В слот-бар приходят клиенты. В игральный автомат, типа «однорукий бандит», каждые 5–10 минут опускается монета номиналом 5 рублей. Автомат случайным образом в течение 10 секунд выдает три цифры от 0 до 9.

В случае совпадения всех трех цифр, игрок выигрывает 50 монет (по 5 рублей), в случае выпадения любой другой комбинации – монета игрока уходит в доход казино. Принятые монеты автомат упаковывает в пачки по 10 штук в каждой.

В случае выигрыша игрок опускает в автомат дополнительно от 10 до 15 монет, на каждую монету он тратит 20 секунд.

Смоделировать работу автомата в течение 24 часов. Определить сумму денег, выигранных игроками, и чистую прибыль казино в рублях.

6. Ремонт автомобилей

Участок ремонта кузовов автомобилей состоит из двух рабочих мест: первое рабочее место – это кузовной ремонт автомобиля, второе рабочее место – окраска кузова. После восстановления кузова автомобили поступают в окрасочную камеру.

Время поступления на ремонт поврежденных автомобилей первой модели – случайная величина, равномерно распределенная на интервале от 1 до 6 часов, второй модели – от 1 до 2 часов. На кузовной ремонт автомобилей первой модели тратится от 1 до 3 часов, второй модели – от 2 до 5 часов. Время окраски любого автомобиля равномерно распределено на интервале (15 – 20) минут.

Модели первого типа при обслуживании имеют более высокий приоритет.

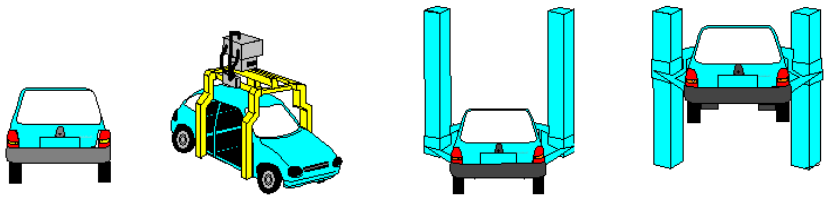
В случае если ремонтная мастерская и покрасочная камера заняты, автомобили дожидаются обслуживания в очередях, длины которых не ограничена.

За 12 часов оценить отдельно для первой и второй модели:

– среднее время, которое тратится на ремонт автомобилей;
– среднее время ожидания в очередях;
– количество отремонтированных автомобилей;
– максимальный размер очереди «ожидания» начала обслуживания и очереди перед операцией окраски.

Проанализировать зависимость приведенных выше характеристик при изменении их числовых значений.

Сделать анимацию в модели: создать картинки для автомобилей (например, для первой модели Entity Picture = Track, для второй - Entity Picture = Van) и ресурсов в состоянии свободен и занят (рис. П1.3).



а *б* *в* *г*

Рис. П1.3. Анимационная картинка ресурсов «Сотрудник 1»:
а – «Первое рабочее место» свободно; *б* - «Первое рабочее место» занято,
в – «Второе рабочее место» свободно; *г* - «Второе рабочее место» занято

7. Полеты рейсовых самолетов

Создать модель полета рейсовых самолетов.

Клиенты, желающие приобрести билет на самолет, приходят в кассу аэропорта в среднем через 20±5, чаще 10 минут, причем 25 % из них приобретают билеты в первый

класс, 70 % – во второй класс, а остальные вообще отказываются приобретать билеты и уходят.

Время вылета самолета определяется его полной загрузкой, т. е. самолет вылетит только при наличии 10 пассажиров первого класса и 20 пассажиров второго класса. Самолеты прибывают в аэропорт в среднем раз в 6–12 часов, максимальное количество самолетов – 20. Время полета занимает в среднем (5±3) часов, чаще 6 часов. По прилету пассажиров отвозят в здание аэропорта, а самолет – на техническое обслуживание.

8. Обслуживание покупателей в магазине

В магазин за покупками приходят клиенты. Для работников магазина клиенты классифицируются на постоянных и обычных. Продавцы (менеджеры) затрачивают в среднем 2 минуты на человека для разъяснения информации по товарам и ответа на вопросы. Приоритетное право на обслуживание без очереди имеют постоянные клиенты; 25 % посетителей уходят без покупок, а остальные встают в очередь в кассу. Кассир один, он обслуживает из очереди постоянных клиентов, а потом обычных посетителей, причем как только кассир рассчитал одного клиента, он сразу же обслуживает следующего, время обслуживания клиента занимает 5 минут.

Постоянные клиенты в основном приходят утром (с 9 до 11 часов) и в конце рабочего дня (с 16 до 18 часов), а обычные посетители – в основном в середине дня.

Построить график, отображающий уровень посещаемости магазина покупателями, и график загруженности кассира.

9. Обработка писем на почте

Люди приносят на почту письма, которые могут быть двух видов: заказные и обычные. Затем почтовые работники их обрабатывают. Заказные письма поступают круглосуточно раз в 5–20 минут, а простые письма принимаются только с 8.00 до 20.00 (8.00–12.00 их количество увеличивается от 50 до 120, наибольшее их количество (260) поступает между 12.00 и 14.00, а затем их количество плавно убывает от 180 до 70).

Оба вида писем обрабатываются одним работником почтовой службы, причем заказные письма обрабатываются вне очереди, т. к. они важнее. Время обработки заказных писем – 1–3 минуты, а время обработки простых писем – 3–5, чаще 4 минуты.

Затем все эти письма поступают в отдел подготовки к отправлению, где заказные письма обслуживаются также вне очереди. Время подготовки писем к отправке – 10–15 минут.

Создать анимацию работы сотрудников почты и отразить процесс обработки простых писем на гистограмме.

10. Обработка шестерен

На участке термической обработки выполняются цементация и закаливание шестерен, поступающих через 10 ± 5 мин.

Цементация занимает 10 ± 7 мин., а закаливание - 10 ± 6 мин. Качество определяется суммарным временем обработки.

Шестерни со временем обработки:

– больше 25 мин. – покидают участок;
– от 20 до 25 мин. – передаются на повторную закалку;
– меньше 20 мин. – должны пройти повторную полную обработку.

Детали с суммарным временем обработки меньше 20 мин. считаются вторым сортом.

Смоделировать процесс обработки на участке 400 шестерен.

Определить:

– количество обработанных деталей;
– число повторений полной и частичной обработки.

Приложение 2

Описание модулей панели основных процессов (Basic Process Panel)

Схемные модули (Flowchart Modules)

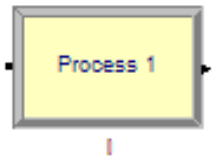
Модуль Create - является отправной точкой для сущностей в имитационной модели. Сущности – это индивидуальные элементы, обрабатываемые в системе. Создание сущностей модулем происходит по расписанию или же основываясь на значении времени между прибытиями сущности в модель. Покидая модуль, сущности начинают обрабатываться в системе. Тип создаваемых сущностей определяется в этом модуле.

Применение: прибытие различных документов в сфере бизнеса (например: заказы, чеки, документация); прибытие клиентов в сфере обслуживания (например: в ресторан, в магазин); начало изготовления продукции на производственной линии.

Таблица П2.1

Параметры модуля Create

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Entity Type	Название типа сущности, который будет создаваться модулем
Type	Способ формирования потока прибытия. Type может иметь значения: *Random* (используется экспоненциальное распределение со средним значением, определенным пользователем), *Schedule* (определяется модулем Schedule), *Constant* (будет использоваться постоянное значение, определенное пользователем) или *Expression* (поток прибытия будет формироваться по определенному выражению)
Value	Определяет среднее значение времени между прибытиями сущностей
Schedule Name	Имя расписания, которое определяет характер прибытия сущности в систему
Expression	Этот параметр задает тип распределения или любое выражение, определяющее время между прибытиями сущностей в модель
Units	Единицы измерения времени между прибытиями (день, час, минута, секунда)
Entities per arrival	Количество сущностей, входящих в систему за одно прибытие
Max arrivals	Максимальное число сущностей, которое может создать этот модуль (ресурс генератора)
First Creation	Время, через которое прибудет первая сущность в модель, от начала моделирования



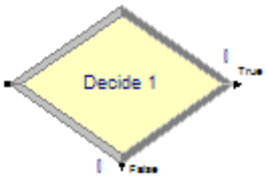
Модуль Process - является основным модулем процесса обработки сущностей в имитационной модели. В модуле имеются опции использования ресурсов, т. е., как и при любой обработке, захватываются какие-то ресурсы. Кроме стандартного модуля Process, можно использовать подмодель, придавая ей особую, определенную пользователем, иерархическую логическую схему. В модуле можно также задавать добавочные стоимостные и временные характеристики процесса обработки сущности.

Наиболее частое применение модуля Process: проверка документов; выполнение заказов; обслуживание клиентов; обработка деталей.

Таблица П2.2

Параметры модуля Process

Параметры	**Описание**
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Type	Определяет логическую схему модуля. *Standard* означает, что логическая схема находится внутри модуля и зависит от параметра Action. *Submodel* показывает, что логическая схема будет находиться ниже в иерархической модели. Подмодель может содержать любое количество логических модулей
Action	Тип обработки, происходящей внутри модуля, может быть четырех типов: *Delay* просто показывает, что процесс занимает какое-то время и не отражает использование ресурсов; *Seize Delay* указывает на то, что в этом модуле были размещены ресурсы и будет происходить их захват и задержка, ресурсы будут захватываться (т.е. будут заняты обработкой сущности), а их освобождение будет происходить позднее с помощью какого-то другого модуля; *Seize Delay Release* указывает на то, что ресурсы были захвачены, а затем (через время) освободились, и *Delay Release* означает, что ресурсы до этого были захвачены сущностью, а в таком модуле сущность задержится и освободит ресурс. Все эти параметры доступны только тогда, когда Type = Standard
Priority	Значение приоритета модулей, использующих один и тот же ресурс где угодно в модели. Это свойство не доступно, если Action = Delay (или Delay Release) или когда Type = Submodel
Resources	Определяет ресурсы или группы ресурсов, которые будут обрабатывать сущности в этом модуле
Delay Type	Тип распределения или процедура, определяющая параметры задержки
Units	Единицы измерения времени задержки (день, час, минута, секунда)
Allocation	Определяет стоимостные характеристики обработки. Value Added – означает учитывать стоимостные характеристики, а Non-Value Added – не учитывать
Minimum	Поле, определяющее минимальное значение для равномерного и треугольного распределения
Maximum	Поле, определяющее максимальное значение для равномерного и треугольного распределения
Value	Поле, определяющее среднее значение для нормального и треугольного распределения или значения для постоянной временной задержки
Std Dev	Параметр, определяющий стандартное отклонение для распределения
Expression	Поле, в котором задается выражение, определяющее значение временной задержки, если Delay Type = Expression



Модуль Decide позволяет описать и задать логику модели, учитывая принятие решений. Он включает опции принятия решений, основанных на условии By Condition или основанных на вероятности By Chance. Условия могут быть основаны на значении атрибута Attribute, значении переменной Variable, типе сущности Entity Type или основанные на выражении Expression.

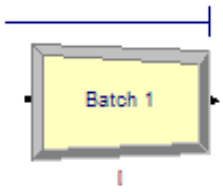
Если поставленное условие выполняется, то сущности будут покидать модуль через ветку True, иначе – по ветке False. Данный модуль позволяет выполнять проверку не только одного условия, но и нескольких. Это достигается с помощью свойства Type → N-way by Chance/by Condition. В зависимости от условия сущность идет по нужной ветке. Таким образом, по ветке True у модуля может быть любое количество выходов (по ветке False – всегда один выход).

Применение: разделение дел на срочные дела и несрочные; перенаправление недоделанных или сделанных неправильно работ на доработку.

Таблица П2.3

Параметры модуля Decide

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Type	Тип принятия решения: *By Chance* – выбор направления основывается на вероятности и *By Condition* – проверка на выполнение конкретно заданного условия
Percent True	Значение, определяющее процент сущностей, который пойдет по направлению True
If	Тип условия, которое будет проверяться на выполнение
Named	Имя переменной, атрибута или типа сущности, который будет проверяться при входе сущности в модуль
Is	Математический знак условия, например больше, меньше, равно и т. д.
Value	Значение, с которым будет сравниваться атрибут или переменная пришедшей сущности. Если тип условия – Expression, то в выражении должен стоять знак условия, например Color<> Red



Модуль Batch отвечает за механизм группировки сущностей в имитационной модели. Группировка может быть постоянной или временной. Временно сгруппированные комплекты сущностей позднее могут быть разъединены с помощью модуля Separate. Комплекты могут состоять из любого числа входящих сущностей, определенного пользователем, или же сущности могут объединяться в комплект в зависимости от атрибута сущности. Временные и стоимостные характеристики выходящей сущности, представляющей комплект, будут равны сумме характеристик вошедших в группу сущностей.

Сущности прибывают в модуль, становятся в очередь и остаются там до тех пор, пока в модуле не будет набрано заданное количество сущностей. Когда соберется нужное число сущностей, создается сущность, представляющая комплект.

Применение: собрать необходимое количество данных, прежде чем начинать их обработку; собрать ранее разделенные копии одной формы; соединить пациента и его больничную карту приема к врачу.

Таблица П2.4

Параметры модуля Batch

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Type	Способ группировки сущностей может быть: *Temporary* (временная) и *Permanent* (постоянная)
Batch Size	Число сущностей, образующих один комплект
Rule	Определяет, по какому признаку будут группироваться. Если Rule = Any Entity, – это значит, что первые 3 (если Batch Size = 3) сущности будут сгруппированы. Если Rule = By Attribute, то будет объединяться заданное количество сущностей с определенным атрибутом. Например, если Attribute Name = Color, то все сущности, имеющие одинаковое значение атрибута Color, будут сгруппированы
Attribute Name	Имя атрибута, по значению которого будут группироваться сущности

Модуль Separate может использоваться в двух возможных антах:

1. Для создания копий входящих сущностей. Если модуль создает копии сущностей, то пользователь может задать количество дубликатов сущности. У дублированной сущности значения атрибута, а также анимационная картинка такие же, как и у оригинала. Оригинальная сущность также покидает модуль.

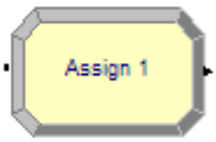
2. Для разделения ранее сгруппированных сущностей. Правило для разделения стоимостных и временных характеристик копий сущностей и разделенных сущностей определяется пользователем. Когда временно сгруппированные сущности прибывают в модуль, они раскладываются на составные сущности. Сущности покидают модуль в той же последовательности, в которой они добавлялись в комплект.

Применение: разъединение ранее сгруппированных комплектов документов; для параллельной обработки счетов и документов по одному заказу.

Таблица П2.5

Параметры модуля Separate

Параметры	Описание
Name	Уникальное имя модуля
# of Duplic	Количество создаваемых копий входящей сущности
Type	Способ разделение входящей в модуль сущности. *Duplicate Original* – просто делает дубликаты входящей сущности. *Split Existing Batch* проводит разгрупппировку
Allocation Rule	Метод разделения стоимости и времени, если выбран Type=Split Existing Batch. Retain Original Entity Values сохраняет оригинальные значения сущностей. Take All Representative Values – все сущности принимают одинаковое значение. Take Specific Representative Values – сущности принимают специфическое значение



Модуль Assign предназначен для задания нового значения переменной, атрибуту сущности, типу сущности, анимационной картинке сущности или другой переменной в системе.

В одном модуле можно сделать любое количество назначений: сменить тип сущности, ее картинку, задать любое количество переменных и т. д.

Пример применения модуля Assign: установление приоритета для клиентов; присвоение номера вышедшему приказу.

Таблица П2.6

Параметры модуля Assign

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Type	Тип назначения, которое будет осуществляться. Other может включать в себя встроенные переменные, такие, как вместимость ресурса или конечное время моделирования
Variable Name	Имя переменной, которая будет изменяться в этом модуле
Attribute Name	Имя атрибута, который будет изменяться в этом модуле
Entity Type	Новый тип сущности, присваиваемый сущности в этом модуле
Entity Picture	Новая анимационная картинка для сущности, прошедшей этот модуль
New Value	Присваиваемое новое значение для атрибута, переменной

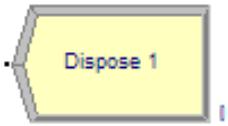
Модуль Record предназначен для сбора статистики в ной модели. Модуль может собирать различные типы статистики, включая время между выходами сущностей из модуля, статистику сущности (время цикла, стоимость), статистику за период времени (период времени от заданной точки до текущего момента). Также доступен количественный тип статистики.

Частое применение модуля: подсчитать, какое количество заказов было выполнено с опозданием; подсчитать количество работы, совершаемое за один час.

Таблица П2.7

Параметры Модуль Record

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Type	Определяет тип статистики, которая будет собираться. *Count* будет увеличивать или уменьшать статистику на заданное значение. *Entity Statistics* будет собирать общую статистику о сущности, например: время цикла, стоимостные характеристики и т. д. *Time Interval* будет считать разницу между значением атрибута и текущим временем моделирования. *Time Between* будет отслеживать время между вхождением сущностей в модуль. *Expression* будет просто фиксировать значение, определяемое выражением
Attribute Name	Имя атрибута, значение которого будет использоваться для интервальной статистики
Value	Значение, которое будет добавляться к статистике, когда в модуль будет прибывать сущность



Модуль Dispose является выходной точкой из имитационной модели. Статистика о сущности может собираться до того момента, пока она не выйдет из системы.

Применение: окончание бизнес-процесса; клиенты покидают отдел.

Таблица П2.8

Параметры модуля Dispose

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Record Entity Statistics	Определяет, будет ли вестись статистика о выходе сущности из системы

Модули данных (Data Modules)

Модуль Entity определяет тип сущности и ее анимационную картинку в имитационном процессе, также определяет стоимостную информацию. Для каждого источника должен быть определен тип сущности, который он генерирует.

Применение модуля Entity: документы (факсы, письма, отчеты и т. д.); люди в моделях больницы или магазина.

Таблица П2.9

Параметры модуля Entity

Параметры	Описание
Entity Type	Название типа сущности
Initial Picture	Графическое представление сущности в начале имитационного процесса. Это значение может быть впоследствии изменено с помощью модуля Assign. Просмотреть анимационные картинки можно так: Edit/ Entity picture

Модуль Queue предназначен для изменения правила расстановки сущностей в очереди, т. е. задается правило обслуживания сущности в процессе. По умолчанию тип очереди First in First out.

Применение: стопка документов, ожидающих освобождения ресурса; место для собирания частей, ожидающих упаковки (группировки).

Таблица П2.10

Параметры модуля Queue

Параметры	**Описание**
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Attribute Name	Имя атрибута, значение которого будет учитываться, если тип = Lowest Attribute Value или Highest Attribute Value
Type	Правило расстановки сущностей в очереди: *First in First out* – первый вошел, первый вышел; *Last in first out* – последний пришел, первый вышел; *Lowest Attribute Value* – первый выйдет из очереди тот, значение атрибута у которого низшее; *Highest Attribute Value* – первый выйдет из очереди тот, значение атрибута у которого наивысшее

Модуль Resource предназначен для определения ресурсов и их свойств в имитационном процессе; кроме того, модуль включает в себя стоимостную информацию о ресурсах и вместимость ресурсов. Ресурсы могут иметь фиксированную вместимость или же основанную на расписании. У ресурсов с фиксированной вместимостью в течение имитационного процесса вместимость изменяться не может. Ресурс должен быть связан с каким-либо процессом.

Применение: люди (клерки, продавцы, бухгалтеры, рабочие и т. д.); оборудование (телефонная линия, станок, компьютер).

Таблица П2.11

Параметры модуля Resource

Параметры	**Описание**
Name	Имя ресурса
Type	Метод, определяющий вместимость ресурса. *Fixed Capacity* – фиксированная вместимость ресурса. *Based on Schedule* – вместимость ресурса определяется модулем Schedule
Capacity	Число ресурсов, находящихся в системе
Schedule Name	Имя Schedule модуля, который определяет вместимость ресурса, если Type = Based on Schedule
Busy / Hour	Почасовая стоимость обработки сущности ресурсом. Время учитывается только тогда, когда ресурс занят обработкой и прекращает учитываться, когда ресурс освобождается
Idle / Hour	Стоимость ресурса, когда он не занят
Per Use	Стоимость обработки ресурсом одной сущности (не зависит от времени)

Модуль Schedule может использоваться вместе с модулем Resource для определения вместимости ресурса и с модулем Create – для задания расписания прибытия сущностей.

Применение: расписание работы персонала с перерывами на обед; значение покупателей, прибывающих в супермаркет.

Таблица П2.12

Параметры модуля Schedule

Параметры	Описание
Name	Название расписания
Type	Тип расписания, который может быть *Capacity* (расписание для ресурсов), *Arrival* (для модуля Create) или *Other* (разнообразные временные задержки или факторы)
Time Units	Масштаб оси времени в графике расписания

Модуль Set, который описывает группу ресурсов, использующихся в модуле Process. В группе могут находиться несколько ресурсов. Модуль Set автоматически создает ресурсы, вместимость которых по умолчанию равна 1, и без всякой стоимостной информации. Следовательно, если для ресурсов, входящих в группу, не нужно стоимостной информации и вместимость более 1, то можно обойтись созданием только модуля Set.

Возможно применение модуля для организации работы группы работников, например по очереди.

Таблица П2.13

Параметры модуля Set

Параметры	Описание
Name	Название группы
Members	Перечисляет ресурсы, входящие в группу. Порядок перечисления ресурсов важен, когда в модуле Process используется правило выбора Cyclical или Preferred Order
Resource Name	Названия ресурсов, входящих в группу

Модули Variable и Attribute определяют значение переменных. Переменные, относящиеся к модулю Decide или Assign, могут использоваться в выражениях. Если переменная не описана в этих модулях, то ее первоначальное значение равно 0.

Применение: число документов обрабатываемых в час; присвоение серийного номера для идентификации продукции.

Таблица П2.14

Параметры модулей Variable и Attribute

Параметры	Описание
Name	Имя переменной
Initial Value	Первоначальное значение переменной. Это значение впоследствии может меняться модулем Assign
Rows	Число строк в размерной переменной
Columns	Число столбцов в размерной переменной
Clear Option	Определяет время, когда значение переменной сбрасывается в начальное значение. *Statistics* – сбрасывает переменную в начальное значение в любой момент, когда статистика была расчищена. *System* – сбрасывает переменную в начальное значение в любой момент, когда система была расчищена. *None* – никогда не сбрасывает переменную в начальное значение, исключая предшествующую первой репликации
Statistics	Определяет, будет ли вестись статистика по этой переменной